SYBASE®
An SAP® Company

System Administration

Sybase Unwired Platform 2.0

DOCUMENT ID: DC01205-01-0200-03
LAST REVISED: August 2011

Sybase, Inc., One Sybase Drive, Dublin, CA 94568.

Contents

CHAPTER 1 Documentation Roadmap for Unwired Platform

Learn more about Sybase® Unwired Platform documentation.

Table 1. Sybase Unwired Platform Documentation

Document	Description
Sybase Unwired Platform Installation Guide	Describes how to install or upgrade Sybase Unwired Platform. Check the *Sybase Unwired Platform Release Bulletin* for additional information and corrections. Audience: IT installation team, training team, system administrators involved in planning, and any user installing the system. Use: during the planning and installation phase.
Sybase Unwired Platform Release Bulletin	Provides information about known issues, and updates. The document is updated periodically. Audience: IT installation team, training team, system administrators involved in planning, and any user who needs up-to-date information. Use: during the planning and installation phase, and throughout the product life cycle.
New Features	Describes new or updated features. Audience: all users. Use: any time to learn what is available.
Fundamentals	Describes basic mobility concepts and how Sybase Unwired Platform enables you to design mobility solutions. Audience: all users. Use: during the planning and installation phase, or any time for reference.

Document	Description
System Administration	Describes how to plan, configure, manage, and monitor Sybase Unwired Platform. Use with the *Sybase Control Center for Sybase Unwired Platform* online documentation. Audience: installation team, test team, system administrators responsible for managing and monitoring Sybase Unwired Platform, and for provisioning device clients. Use: during the installation phase, implementation phase, and for ongoing operation, maintenance, and administration of Sybase Unwired Platform.
Sybase Control Center for Sybase Unwired Platform	Describes how to use the Sybase Control Center administration console to configure, manage and monitor Sybase Unwired Platform. The online documentation is available when you launch the console (**Start** > **Programs** > **Sybase** > **Sybase Control Center**, and select the question mark symbol in the top right quadrant of the screen). Audience: system administrators responsible for managing and monitoring Sybase Unwired Platform, and system administrators responsible for provisioning device clients. Use: for ongoing operation, administration, and maintenance of the system.
Troubleshooting	Provides information for troubleshooting, solving, or reporting problems. Audience: IT staff responsible for keeping Sybase Unwired Platform running, developers, and system administrators. Use: during installation and implementation, development and deployment, and ongoing maintenance.

Document	Description

Tutorials	Tutorials for trying out basic development functionality.
	Audience: new developers, or any interested user.
	Use: after installation.
	• Learn mobile business object (MBO) basics, and create a mobile device application:
	  • *Tutorial: Mobile Business Object Development*
	• Create native mobile device applications:
	  • *Tutorial: BlackBerry Application Development*
	  • *Tutorial: iOS Application Development*
	• Create a mobile workflow package:
	  • *Tutorial: Mobile Workflow Package Development*

Sybase Unwired WorkSpace – Mobile Business Object Development	Online help for developing MBOs. Audience: new and experienced developers. Use: after system installation.
Sybase Unwired WorkSpace – Mobile Workflow Package Development	Online help for developing mobile workflow applications. Audience: new and experienced developers. Use: after system installation.
Developer guides for device application customization	Information for client-side custom coding using the Client Object API. Audience: experienced developers. Use: to custom code client-side applications. • *Developer Guide for BlackBerry* • *Developer Guide for iOS* • *Developer Guide for Mobile Workflow Packages* • *Developer Guide for Windows and Windows Mobile*

Document	Description
Developer guide for Unwired Server side customization – *Developer Guide for Unwired Server*	Information for custom coding using the Server API. Audience: experienced developers. Use: to customize and automate server-side implementations for device applications, and administration, such as data handling. Dependencies: Use with *Fundamentals* and *Sybase Unwired WorkSpace – Mobile Business Object Development*.
Developer guide for system administration customization – *Developer Guide for Unwired Server Management API*	Information for custom coding using administration APIs. Audience: experienced developers. Use: to customize and automate administration at a coding level. Dependencies: Use with *Fundamentals* and *System Administration*.

CHAPTER 2 Administration of the Unwired Platform

At its heart, Unwired Platform is a mobility-enablement platform. It has the tools, the client APIs, the server components and the administration console that offer a complete, end-to-end system for creating enterprise-level mobile solutions.

Administrators interact with Unwired Platform primarily to configure platform components and ensure the production environment works efficiently as a result of that configuration.

Unwired Platform delegates security checks, by passing login and password information to the security provider. In general, all administrative and application users and their passwords are managed in the security repository. Sybase Control Center limits feature visibility depending on the role an administrator logs in with. Unwired Platform administrators can be one of two types, each with distinct logins:

- Unwired Platform administrator (also known as the platform administrator) – has cluster-wide administration access to the Unwired Platform. supAdmin is the default login for cluster-side administration and is assigned the "SUP Administrator" role in OpenDS (the default Unwired Platform repository for development environments). In a deployment edition, you must map the SUP Administrator logical role to a role in your existing repository.
- Domain administrator – has access confined to the specific domains that the platform administrator assigns. supDomainAdmin is the default login for domain administration and is assigned the "SUP Domain Administrator" role in OpenDS. In a deployment edition, you must also map SUP Domain Administrator role to a role in your existing repository.

Both types of administrators are authenticated and authorized based on the 'admin' security configuration and its domain-level role mapping for administrative roles.

There are three main aspects to this platform that platform and domain administrators conjointly administer:

- Mobile application creation is supported by integrated development tools, APIs, samples and tutorials. For the developer, this aspect of the Unwired Platform allows the creation of integration logic and device applications that allow registered device users to seamlessly interact securely with your existing back-end infrastructure.

  Before you begin, know: the devices you need to support, the back-ends you need to integrate with, and the synchronization model you will use.
- Mobile application administration requires both the development and deployment of applications. The Unwired Platform perspective in Sybase Control Center is integral to configuring and managing applications as they are developed and integrated into your

system landscape. Further, all aspects of a production environment can be monitored for troubleshooting or performance tuning purposes.

Before you begin, decide: the system design/topology you environment requires and what type of security providers you need to delegate application security to.

- Mobile user, device, and application management simplifies how the end user is registered and provisioned in the Unwired Platform environment. When Afaria® is used in conjunction with Unwired Platform, an administrator has a powerful cross-platform mobile management framework:
  - Sybase Control Center performs the package deployment to the Unwired Server as well as manages user accounts and data service subscriptions.

  Before you begin, understand what package types you need to support, and how the package type affects how users are registered and subscriptions are created, and how your devices might be provisioned (cable or over-the-air).

CHAPTER 3 Deployment Architecture Planning

Plan your deployment by mapping the logical architecture of Unwired Platform components to a physical computing environment (new or existing computing nodes).

Most customers' needs are met by using one of these environment types: single node environments (sometimes known as trial environments), shared development and test environments, simple redundancy environment, full-scale (or optimal) redundancy environments, and even multitenant environments (for hostability).

However, there are common characteristics to all of them. These characteristics most often include:

- A discrete network architecture that separates production, test, and development environments.
- The isolation of Internet-facing systems from internal systems.
- The configuration of a new or an existing authentication and identity management solution for internal and remote proof-of-identity and access control.
- An effective and efficient redundancy plan — except in the case of single node installations.

Environment Options

Different Unwired Platform environments are designed to support different life cycle stages and whether you are hosting that environment specifically for your enterprise or to host multiple tenants.

This table summarizes the options you can design:

Environment	Purpose	Supports multi-node clusters?	Supports multiple tenants?
Single-node	To install a trial version of Unwired Platform To install all components on a single machine as part of a personal development license.	No, only a single-node installation.	Yes

Environment	Purpose	Supports multi-node clusters?	Supports multiple tenants?
Shared development	To support multiple developers with multiple developing tooling installations on individual computers but share runtime component on the network (that is, servers and runtime databases).	Yes, though typically not needed	Yes, though typically not needed
Fault-tolerant production or pre-production (test) systems	To create a system based on real requirements of an organization and its users.	Yes	Yes

Clusters

As an organization grows, Unwired Platform administrators need to create a scalable IT infrastructure using clusters. Clustering creates redundant Unwired Platform components on your network to provide a highly scalable and available system architecture.

Organizations can seamlessly achieve high availability and scalability by adding more or redundant instances of core components. Redundant instances of critical components provide transparent failover.

In a production environment, the Unwired Platform deployment typically uses at least one relay server. The connections to relay servers can be configured within a cluster instance from Sybase Control Center.

Nodes and Cluster Size

A node is a host upon which one or more components have been installed. A cluster consists of Unwired Platform components running on one or more nodes that work together as a single, continuously available, system in order to provide seamless application management, device management, and data access to users.

Each node on a cluster is a fully functional part of the unwired system. In a clustered environment, the nodes work together to provide increased availability and performance.

Hosts in a cluster should have similar processing, memory, and I/O capability to enable load balancing without significant performance degradation. Hosts in a data tier cluster should be more powerful than hosts in an Unwired Server cluster.

There are different node strategies you can employ, depending on the environment you are designing:

- **Single-node installations for personal development or trial installations –** A nonredundant architecture consisting of an Unwired Server and data tier installed on a single host.
- **2-node installations for enterprise development and testing –** A primitive architecture without load balancing that may optionally use a relay server. The data tier (which includes the CDB, the messaging database, and the monitoring database) is on one host and the server tier (which includes Unwired Server and Sybase Control Center) is on another.
- **3-node Unwired Server cluster for entry level production environments –** A simple redundant architecture with two server tier nodes, and a separate data tier host. For an example of this cluster, see *Redundant Architecture Options*.
- **N+2-node Unwired Server clusters and data tier clusters for full-scale production environments –** An optimally redundant architecture with any number of server tier nodes, supported by a relay server and a data tier cluster. For an example of this cluster, see *Redundant Architecture Options*.

Cluster Types

Clusters are groups of similar components that work together to service client requests.

A cluster is a parallel or distributed computing system made up of many discrete hosts that form a single, unified computing resource. Through clustering, you can partition the system load across redundant Unwired Platform components to design a highly-available system.

There are two tiers you can install in a cluster: the server tier (Unwired Server) and the data tier (runtime databases). The server tier may further be supported by a relay server and a load balancer, depending on the scale of your rollout.

Each tier uses a different type of cluster model:

- Load-balancing server tier clusters – improve the system performance by sharing workloads and requests, and improving the efficiency of Unwired Server services (like synchronization and deployment). Requests initiated from the client are managed by a load balancer (or application request routing services in the case of some IIS deployments) for two or more relay servers. This load balancer distributes requests to these relay servers (or some other mechanism) among all the Unwired Server cluster nodes. Each node in the Unwired Server cluster scales independently and automatically when another Unwired Server node joins or leaves the cluster.
- Failover data tier clusters – improve the availability of runtime database services to Unwired Servers. Failover clusters have a redundant node, which provides data tier services when the primary node fails. The most common size for a failover consolidated database cluster is two nodes, which is the minimum requirement to provide redundancy and thereby eliminate single points of failure.

  Note: The data tier does not need a relay server, since it will not be accessed directly from outside the firewall.

Each Unwired Server cluster has a primary node:

- The primary node contains the master copy of the configuration repository for all nodes in the cluster. The primary node distributes its configuration to the other nodes. When a primary node fails, a new primary is elected from the remaining nodes.
- Each secondary node gets its configuration from the primary node. Nodes must have unique names, but are identified as members of the same cluster as the primary node.

Each Unwired Server in a cluster runs on a separate host. It can connect to the data tier independently, as well as have its own copy of the system files required for execution. Sybase recommends that each cluster node run from its own installation directory.

Load Balancing

Load balancing is a high-availability strategy that prevents any one host from getting overloaded with work, thereby adversely affecting overall system performance.

Load balancing supports the server tier. Achieve improved performance by balancing loads. Do this by:

- Creating and configuring two or more Unwired Servers in a cluster.
- Using some form of a centralized task distribution mechanism (for example, a relay server).
- Optionally using a third-party load balancer, with multiple relay servers. If you use a load balancer, Sybase recommends that you use either an intelligent layer 4 switch balancer (with dead server detection enabled), Microsoft Network Load Balancing cluster, or Microsoft IIS Application Request Routing (recommended). For either type, ensure that the same wildcard certificate is installed on all Unwired Server nodes, so the device client needs the load balancers hostname configured. For information on wildcard certificate setup, see your load balancer's documentation. For information on how to configure your relay server environment to use one of the supported Windows load balancers, see *Enabling Application Request Routing for IIS Relay Server Hosts*.

See also

- *Enabling Application Request Routing for IIS Relay Server Hosts* on page 69

Unwired Server Clusters

An Unwired Server cluster provides highly available services to remote-device clients, typically requiring one or more relay servers and an optional load balancer.

When you install Sybase Unwired Platform, a cluster is created automatically, and Unwired Server is added to it.

By default, the cluster name is the name of the first host on which you install Unwired Server. Each server can be a member of only one cluster. Cluster configuration properties are saved in the cluster database, which is managed by the data tier.

For high availability, at least two servers must be defined for a cluster. However, to create a load-balanced system, many clients then also use a relay server. Without a relay server, clients require custom logic to determine which Unwired Server will directly receive requests.

However, a relay-server free environment is not a highly available environment, nor can requests be distributed equally across multiple servers. Consequently, Sybase recommends that you always use a relay server.

See also

- *Chapter 5, Relay Server Clusters* on page 53
- *Implementing an N+2-node Cluster for Optimal Redundancy* on page 14

Failover

Failover is another high-availability strategy that allows a secondary node to take over in the event of a fault or failure in the first node, thus allowing normal use to continue. Automatic failover is a default behaviour of the cluster; it does not require manual intervention.

Failover supports the Unwired Platform data tier, which includes the consolidated database, cluster database, messaging database, and monitoring database.

Fault-tolerance of the data tier through failover uses a passive/active node configuration where only one node is active at any given moment (that is, if one node fails, another standby node becomes active). Consequently, administrators must ensure that their data storage is protected via a redundant array of independent disks (RAID) cluster.

Data Tier Clusters

The data tier can be clustered to make system data used by the server tier more fail-safe.

When creating a cluster for the data tier, administrators must use Microsoft Cluster Service (or Failover Cluster) to ensure data fails over and thereby enhances data reliability by ensuring that does not have a single point of failure. Microsoft Cluster Service (or Failover Cluster) allows Unwired Platform to fail over the data tier when the current server fails. Both servers (active/failed and passive/takeover) must have a connection to the storage subsystem.

The consolidated database data, along with the Advantage messaging database data, should be stored on a redundant array of independent disks (RAID) cluster. You can use any number of disks in a redundant array. However, the more disks you have in the array, the more fail-safe it is; performance degrades with the more disks you introduce. Sybase recommends that you use a RAID 1 disk array for deployments that require a high degree of fault tolerance and performance from their Unwired Platform data tier.

See also

- *Data Management Overview* on page 189
- *Backup and Recovery* on page 264

Shared Development Environments

Mobility projects in medium-to-large organizations usually span multiple development teams, environments, and geographies. In a shared development environment, organizations

typically opt for a single-node environment, but may also choose to use domains to partition the environment.

To allow distributed teams to collaborate successfully, you must take an active role in planning the configuration and management of the environment used to develop mobile applications. Your efforts have a direct bearing on the developer's ability to produce quality applications in a timely manner.

To support collaboration, Sybase recommends that developers:

- Share an Unwired Server, so packages can be deployed to a common server. Encourage application developers to develop deployment packages that allow interface designers and business process designers to download the latest versions of the components.
- Place the Unwired Server and data tier on the same node.
- Use a common system data tier for development and testing so that data is centrally managed.
- Optionally divide developers or groups of developers into "tenants", and use domains to partition the server environment. Make each developer (or a representative developer from development groups) a domain administrator so he or she can deploy packages to the shared Unwired Server.
- Ensure that the development environment is similar to, but separate from, the production environment.

  Even if developers create and test their projects locally, they should still test interactions between system components in an environment that mirrors the production environment. Early testing that simulates the production environment, leads to earlier discovery and correction of security and data-tier-related issues.
- Open LDAP, is, by default, installed with Development Edition licenses. The LDAP directory is already populated with required system roles, which allows you to get basic application security running with minimal investment and effort. However, you may want to re-create the application security environment, so development environments and production environments are similar.

See also

- *Chapter 4, Component Deployment* on page 23
- *Multitenant Environments* on page 19

Fault-Tolerant Production Environments

You can design for fault-tolerance by introducing component redundancy in your environment.

For Unwired Platform installations, Sybase recommends a redundant node and cluster strategy for most production environments. A fault-tolerant system supports:

- Load balancing – uses identical Unwired Platform components on a network to balance the number of requests or tasks among these redundant components in parallel.

- Failover – uses identical Unwired Platform components on a network in order to switch to any of the remaining instances in the event of a component failure.

In an Unwired Platform production environment, system design tends to favor load balancing over failover because of the better efficiency it also yields in rather than just ensuring that the system is more reliable.

Redundant Cluster Architectures

Use a redundant architecture to back up Unwired Platform production deployments with multiple secondary resources to create a fault-tolerant environment using clusters on a server or data tier.

- A simple 3-node cluster is an entry-level redundant architecture used by production deployments.
- An optimal N+2 cluster is the Sybase-recommended production architecture.
- Virtualization allows you to manage system infrastructure on virtual machines. Virtualization allows platform components to run in isolation, but side-by-side on the same physical machine (typically a 64-bit processor with virtual symmetric multiprocessing). Virtualization, while supported in Unwired Platform deployments, is beyond scope of this document.

Simple Redundant Cluster

The simple 3-node cluster requires minmal hardware investment. Consequently, the simple cluster is an attractive alternative for smaller environments.

A simple cluster includes:

- An Unwired Server cluster on the corporate LAN that typically includes two Unwired Servers: a primary server and a secondary server. Each node is installed on its own host node, and would also include a Unified Agent (for remote management via Sybase Control Center)and a relay server outbound enabler (RSOE) (for relay server connections). In Sybase Control Center, you would therefore have a single entry in the Clusters folder and two entries in the Servers folder.
- The Unwired Platform data tier installed to another host on the corporate LAN. The data tier includes all runtime databases: a conoslidated database, a messaging database, and a cluster database, each having their own purpose in supporting the runtime environment.
- A relay server installed to its own host on the DMZ that directs device client connections to one of the two servers in the Unwired Server cluster.

Optimally Redundant Cluster

The optimally redundant cluster is more reliable, multi-node deployment and requires a larger hardware investment because of the number of nodes involved. An optimally redundant cluster is the Sybase-recommended solution for larger user environments.

An optimally redundant cluster includes at least two clusters of different types (server versus data):

- An Unwired Server cluster on the corporate LAN that typically includes two Unwired Servers: a primary server and a secondary server. Each node is installed on its own host node, and would also include a Unified Agent (for remote management via Sybase Control Center)and a relay server outbound enabler (RSOE) (for relay server connections). In Sybase Control Center, you would therefore have a single entry in the Clusters folder and two entries in the Servers folder.
- An data tier cluster on the corporate LAN that typically includes two nodes: a primary node and a failover node. Typically, this is cluster is implemented with Microsoft Disk cluster or a RAID array. The data tier includes all runtime databases: a conolidated database, a messaging database, and a cluster database, each having their own purpose in supporting the runtime environment.
- At least two relay servers installed to distinct hosts on the DMZ. These relay servers can be configured as a relay server cluster if used in conjunction with an optional load balancer or other load balancing technologies.

Implementing an N+2-node Cluster for Optimal Redundancy

An N+2 cluster is the recommended cluster for achieving an optimal level of redundancy in your Unwired Platform architecture.

Prerequisites

Sybase recommends that you have Windows Server 2008 installed on all participating nodes in the cluster, irrespective of whether the node is a runtime server tier node or a data tier node. Windows Server 2003 is supported on non-cluster installations only.

Task

Once you have set up the environment and deployed components, perform end-to-end tests that validate the cluster before you make it publically available.

1. *Setting Up the Microsoft Cluster*

   Set up the Failover Cluster before you install the Unwired Platform data tier.
2. *Setting Up Data Tier Nodes*

   Set up a data tier node using the Unwired Platform installer. The data tier node includes the consolidated database (CDB), cluster database, monitoring database, and messaging database.
3. *Adding a Generic Service to the Data Cluster*

   Before you install Unwired Server, you must add a generic server resource for the data tier cluster.
4. *Setting Up Unwired Server Nodes*

   After you confirm that the generic service is available for the data tier cluster, install and configure the Unwired Server components.
5. *Installing Third-party Software*

If you are using third-party software with Sybase Unwired Platform, copy the third-party software components (DLLs, JARs, and libraries) to each server in a cluster.

6. *Validating the Cluster*

   Once the runtime and data tiers are installed, ensure that these components have been set up correctly.

See also

- *Unwired Server Clusters* on page 10
- *Chapter 5, Relay Server Clusters* on page 53

Setting Up the Microsoft Cluster

Set up the Failover Cluster before you install the Unwired Platform data tier.

Setting up the Failover Cluster may require the assistance of your IT department, especially if you lack the required permissions to complete these tasks.

Note: Perform these steps on all nodes participating in the Failover Cluster unless otherwise indicated.

1. Ensure Failover Clustering is added to the Windows Server 2008 Enterprise Edition.

   Failover Clustering is required on all hosts on which Unwired Platform data tier components will be installed. Failover Clustering is installed by default on Windows Server 2008 Enterprise Edition.

   a) Open the Server Manager console.
   b) Click **Features** in the navigation pane to display the **Feature Summary**.
   c) Below Remote Server Administration Tools, look for Failover Clustering Tools.
   d) If Failover Clustering Tools is not listed, click **Add Features** in the right corner of the **Feature Summary** pane and enable the Failover Clustering feature.

2. Create the Failover Cluster:

   a) Click **Start > Programs > Administrative Tools > Failover Cluster Management**.
   b) Expand **Management** and start the Create Cluster wizard.
   c) On the Access Point for Administering the Cluster page, assign the Unwired Platform cluster name, for example, `SUPdataNode1`.

      Remember this name and use it with the Unwired Platform installer.
   d) Finish creating the cluster and exit the wizard

Next

Install the Unwired Platform data tier components before returning to the console to create a generic service for each node.

Setting Up Data Tier Nodes

Set up a data tier node using the Unwired Platform installer. The data tier node includes the consolidated database (CDB), cluster database, monitoring database, and messaging database.

Note: Perform these steps on all nodes in the Failover Cluster unless otherwise indicated.

1. Run the Sybase Unwired Platform installer.

   Use the installer to set up your enterprise deployment license, installation folder, and so on. See *Sybase Unwired Platform Installation Guide*.
2. When prompted to choose the installation type, choose **Cluster Setup** and click **Next**.
3. When prompted to choose the platform features to install, select **Install data tier for your Unwired Platform cluster**, and click **Next**.

   Continue running the installer as documented in *Sybase Unwired Platform Installation Guide, Installing the Data Tier*.
4. Review the summary and click **Install**.

   The installer creates all required data tier services. For a complete list of services including those names of data services, see *Unwired Platform windows Services*.

   Note: The consolidated database server name must be identical on all data tier hosts in the cluster.

See also

- *Unwired Platform Windows Services* on page 291

Adding a Generic Service to the Data Cluster

Before you install Unwired Server, you must add a generic server resource for the data tier cluster.

1. Open the Failover Cluster Management console.
2. Expand <ClusterName>.sybase.com in the navigation pane, then right-click **Services and Applications > Configure a Service or Application** to start the High Availability wizard.
3. Click **Next** until you reach the **Select Service or Application** page.
4. Select data tier service, and click **Next**.
   - For the consolidated database and cluster database, choose **SybaseUnwiredPlatformConsolidatedDatabase**.
   - For the messaging database, choose **Advantage Database Server**.
5. Name the resource.

   You must use this name as the CDB host name when you set up the Unwired Server cluster.

6. Click **Next** on all other screens to accept the defaults used by the wizard.
7. Click **Finish**.
8. Use the Failover Cluster Management console to verify that you started the resource.

   Contact your IT department if this event message is recorded:

   ```
   Event ID 1194: Cluster network name resource <Name>GenSvc failed
   to create its associated computer object in domain "sybase.com"
   for the following reason: Unable to create computer account.
   ```

Setting Up Unwired Server Nodes

After you confirm that the generic service is available for the data tier cluster, install and configure the Unwired Server components.

Note: Perform these steps on all Unwired Server nodes, unless otherwise indicated.

1. Run the Sybase Unwired Platform installer.

   Use the installer to set up your enterprise deployment license, installation folder, and so on. See *Sybase Unwired Platform Installation Guide*.
2. Choose the node to install.

   The first node of the Unwired Server cluster determines the cluster name by adopting the host name. For subsequent nodes, you need not provide a node name. Each Unwired Server that joins the cluster assumes the cluster name, with an incremented number. For example, the first server is named SybaseUnwiredPlatformMYHOSTServer1, the second SybaseUnwiredPlatformMYHOSTServer2, and so on.

   Note: If the cluster is to be associated with a hosted Relay Server, use only alphanumeric names. Relay Server does not allow any non-alphanumeric character in a server name.
3. When prompted to choose the installation type, choose **Cluster Setup** and click **Next**.
4. When prompted to choose setup type, select either **Install the first server node and connect it to the data tier** or **Install an additional server node and connect it to the data tier**. Provide the Microsoft cluster name (that is, the MS Cluster Access point as the host name of the data tier).

   Continue running the installer as documented in *Sybase Unwired Platform Installation Guide, Installing the Unwired Server*.

Next

In a production environment, you may need to register the server with Sybase Control Center and tune performance properties for the server. See *System Administration > Environment Setup > Server Performance Tuning*.

See also

- *Cluster Administration Overview* on page 141
- *Server Administration Overview* on page 142

Installing Third-party Software

If you are using third-party software with Sybase Unwired Platform, copy the third-party software components (DLLs, JARs, and libraries) to each server in a cluster.

Third-party files is kept separated from Unwired Platform files. There are two reasons for this requirement:

- To isolate third-party from core Unwired Platform files. This can make an upgrade process smoother, since third-party software typically remains untouched.
- To more easily identify the files that need to be applied to a new installation on a new component host machine.

1. Stop Unwired Server.
2. Copy third-party JARs files to `<UnwiredPlatform_InstallDir>\Servers\UnwiredServer\lib\3rdparty`.
3. Copy third-party DLLs to `<UnwiredPlatform_InstallDir>\Servers\UnwiredServer\lib`.
4. Restart Unwired Server.
5. If you are running a cluster, repeat these steps on all server tier nodes.

Validating the Cluster

Once the runtime and data tiers are installed, ensure that these components have been set up correctly.

Prerequisites

Install sampledb on the first node, and ensure the service has started. sampldb is not installed with a deployment edition license. To install sample db, run:

```
<UnwiredPlatform_InstallDir>\Servers\UnwiredServer\Bin
\sampledb.bat install auto
```

Task

End-to-end validation requires that you create a simple MBO and deploy it to Unwired Server. You can use an Unwired Platform tutorial to help you create this package and mobile client.

1. In Unwired Workspace, use a Getting Started tutorial to create a sample MBO and device application that uses data in sampledb.
2. Run the sample application from the emulator.
3. If the synchronization appears to be successful, open the Cluster Management console on your first data tier node, and move the generic service resource you created to the second data tier node.
4. Redeploy your application to the emulator.

Synchronization should work no matter which data tier is used to consolidate platform data from sampledb.

Next

Continue your environment setup and system configuration.

Single-Node Environment

A single-node installation does not make use of node or cluster redundancy. In trial or personal development scenarios, you may want to operate Unwired Platform on a single host.

Unwired Platform treats a single-node cluster, or standalone installation, as a cluster of one member.

This type of environment allows for some degree of administrative convenience in a production environment. Typically, however, you use this model to provide a platform for personal development or IT evaluation of Unwired Platform functionality. Considerations for each are summarized by environment type:

- For personal or enerprise development, a single-node installation uses OpenDS by default to authenticate user access for Unwired Platform. It takes, little or no configuration to make this system functional.
- For production deployments, a noAuth provider is enabled by default, which passes all authentication requests. Administrators must configure a provider immediately after installation to ensure that Unwired Server and Sybase Control Center are secured.

See also

- *Chapter 4, Component Deployment* on page 23

Multitenant Environments

Multitenancy allows platform administrators to deploy a single production environment to service multiple client organizations (known as tenants). Multitenancy uses domains, which allow a tenant's administrators to manage Unwired Platform entities within the cluster partition, provided that administration users possess the proper domain administration privileges.

The platform administrator has a complete view of the enterprise and its clusters, including all domains that are created to support the tenancy strategy. However, domain administrators see only the domains to which they have been assigned. Security configurations can be designed and assigned for a particular domain according to the tenant's security requirements. Likewise, domain-specific packages created by the tenant can be deployed to the domain created for him or her.

Do not confuse domains in Unwired Platform with the traditional concept of network domains. In Unwired Platform, a domain is only a namespace used in production environments.

See also

- *Domain Administration Overview* on page 154

Domains

Domains provide a logical partitioning of a hosting organization's environment that achieves increased flexibility and granularity of control in multitenant environments. By default, the installer creates a single domain named "default."

Administrators use different domains within the same Unwired Platform installation. Domains enable the management of application metadata within a partition, including server connections, packages, role mappings, domain logs, and security, so that changes are visible only in the specific domain.

Considerations when implementing domains in a multitenant environment include:

- Create and manage domains using Sybase Control Center from the Unwired Platform administration perspective of Sybase Control Center.
- You can support multiple customers inside the same Unwired Platform cluster.
- You can configure security specifically for individual domains by creating one or more security configurations in the cluster, and then assigning those security configurations to a domain. You can then map the security configurations to one or more packages. A user accessing the package from a device application is authenticated and authorized by the security provider associated with the package.
- Customers may require their own administrative view on their portion of the Unwired Platform-enabled mobility system. By granting domain administration access to your customers, you can allow customers to customize their deployed applications packages and perform self-administration tasks as needed.

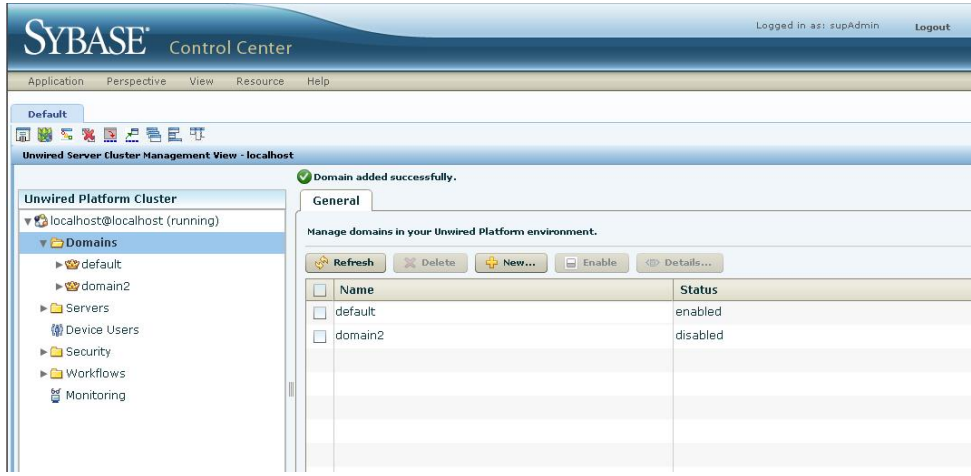
The "default" domain

The "default" domain is a special domain where critical runtime configuration artifacts exist. These artifacts include:

- An "admin" security configuration – this security configuration is mapped to the "default" domain and is used to authenticate and authorize administrative users. For this reason, administrators are not allowed to unassign the "admin" security configuration from the "default" domain.
- Consolidated database (CDB) data source connections – for the "default" CDB data source, users can configure the Pool Size property in the "default" domain according to their requirements. This setting allows the maximum number of open connections to the SQL Anywhere® database server hosting the CDB.

- Monitor database data source connections – the customer can modify the existing monitoring datasource properties according to their configuration requirements, or create a new monitoring datasource in the "default" domain.

Since these critical runtime-related artifacts are located in the "default" domain, administrators are not allowed to delete this domain. Sybase recommends creating new domains to facilitate tenants according to their application requirements.



CHAPTER 4 Component Deployment

How you deploy and set up platform components depends on factors you decided during your deployment planning: what type of environment, how many clusters, how the data tier supports the runtime, how many backends, and so on.

Task	Required by
Distribute components to different hosts on the network using the installer in iterations.	• Single-node –not applicable. • Shared development – Required. See *Sybase Unwired Platform Install Guide > Installing Developer Editions*. • Multinode production – Required. See *Sybase Unwired Platform Install Guide > Installing the Deployment (Production) Edition*.
Enabling administrative logins for Sybase Control Center.	• Single-node – Optional. • Shared development – Suggested. • Multinode production – Required. See *Security Administration*.
Configure and enable security configurations to authenticate device users.	• Single-node – Suggested. • Shared development – Required. • Multinode production – Required. See *Security Administration*.
Configure Unwired Server.	• Single-node – Optional. Defaults should be acceptable. • Shared development – Optional. Only if your environment requires something other than the defaults. • Required for the primary server. Configuration is then distributed to secondary servers. See *Server Environment Administration*.

Task	Required by
Set up relay servers and install relay server outbound enablers (RSOEs) as required.	• Single-node – Optional. In a development environment, you can set up relay server environments to test applications with physical devices. Sybase recommends that you use Sybase Hosted Relay Service with the required number of RSOEs for this purpose. • Shared development – Optional. • Multinode production – Required. See *Relay Server Setup*.
Create and assign security configurations for application security (authentication, authorization, and so on).	• Single-node – Optional. • Shared development – Recommended, to test security features. • Multinode production – Required. See *Security Configurations*.
Create data source connections.	• Single-node – Required. • Shared development – Required. • Multinode production – Required. See *EIS Connections*.
Install third-party software required by some MBO packages to allow them to connect to EIS sources.	• Single-node – Optional. • Shared development – Optional. • Multinode production – Optional. See *Installing Third-Party Software* .
Add Afaria to the Unwired Platform provisioning environment.	• Single-node – Optional. • Shared development – Optional. • Multinode production – Recommended. Afaria automates many provisioning tasks and enhances the security of data stored on the device. Afaria is no longer included with Unwired Platform. You must purchase Afaria licenses seperately and install Afaria components to different computing nodes.

Data Tier

Setting up a database is only required when you are not installing a new consolidated database (CDB) or monitoring database in your production environment.

There are three scenarios that Unwired Platform supports:

- Use only what is installed with the Unwired Platform installer – This is the simplest and recommended scenario; no additional setup is required. You can begin using the databases immediately.
- Use an existing SQL Anywhere installation, but create new database files based on Sybase specifications – This requires that you use a combination of dbinit to initialize the new CDB correctly, and batch scripts that setup the schema for the monitoring database.
- Use an existing SQL Anywhere installation, but use existing database instance – Can be the most risky implementation and of the three is the least recommended option. You must still use the batch scripts to setup the monitoring database schemas; for the CDB, you must use RTRIM to avoid getting incorrectly formatted data on the device. See the SQL Anywhere documentation about using the RTRIM function.

1. *Deploying the CDB*

   Take care when deploying the consolidated database (CDB) in a production environment. The CDB must be installed and configured in such a way as to avoid any adverse impacts on performance. Depending on whether you are using the installed CDB, using an existing one, or initializing a new database file separate from the installer, perform the corresponding task accordingly.

2. *Deploying the Monitoring Database*

   You can use the installed database or an existing database as the monitoring database. Irrespective of which type of database you deploy, you want to set it up in a way that avoids unduly impacting runtime performance of the system.

Deploying the CDB

Take care when deploying the consolidated database (CDB) in a production environment. The CDB must be installed and configured in such a way as to avoid any adverse impacts on performance. Depending on whether you are using the installed CDB, using an existing one, or initializing a new database file separate from the installer, perform the corresponding task accordingly.

See also

- *Deploying the Monitoring Database* on page 28

Initializing a New Consolidated Database

If you are experiencing incorrect string comparisons, you may need to initialize a new database to ensure that comparisons occur correctly. Incorrect string comparisons only occur

when you either use a SQL Anywhere database that has not been installed and set up with the Unwired Platform installer.

Initialize a database using the dbinit utility. This utility is located in `<UnwiredPlatform_InstallDir>\Servers\SQLAnywhere11\BIN32`.

By default, SQL Anywhere creates a database treating a trailing blank as any other character in comparisons during collation. To avoid treating trailing blanks as significant characters, a CDB should be created with the "-b" option that directs SQL Anywhere to ignore trailing blanks in any comparison.

Run dbinit using this syntax: `dbinit -z UTF8 -zn UTF8 -b`

Where:

- -z UTF8 sets the collation sequence data as UTF8.
- -zn UTF8 sets the collation sequence for sorting and comparing to UTF8.
- - b sets up blank padding.

See also

- *Changing the Consolidated Database Server Thread Count and Pool Size* on page 26
- *Using a Different Database and Log Path* on page 27

Changing the Consolidated Database Server Thread Count and Pool Size

You can change the consolidated database (CDB) server thread count and maximum pool size as required. The procedure for this varies depending on the type of environment the CDB server is deployed to.

Sybase has identified two scenarios: one for clustered environments and one for single node installations.

See also

- *Update Properties (updateprops.bat) Utility* on page 354
- *Synchronization Model Performance Tuning* on page 37
- *Initializing a New Consolidated Database* on page 25
- *Using a Different Database and Log Path* on page 27

When data-tier is on remote host in a cluster environment

1. On the data-tier node, remove the service by running:
   ```
   <UnwiredPlatform_InstallDir>\Servers\UnwiredServer\bin
   \install-sup-sqlany11.bat remove
   ```
2. To change the thread count, edit the `<UnwiredPlatform_InstallDir>\Servers\UnwiredServer\bin\install-sup-sqlany11.bat` to modify the CDB_THREADCOUNT property.

Choose an appropriate server threadcount according to your cluster performance requirements.

For example, this command sets the server thread count to 200:

```
set CDB_THREADCOUNT=200
```

3. To change the maximum pool size, edit the `<UnwiredPlatform_InstallDir>\Servers\UnwiredServer\Repository\Instance\com\sybase\djc\sql\DataSource\default.properties` to modify the maxPoolSize property.
4. Recreate the service again with the correct thread count by running:

```
<UnwiredPlatform_InstallDir>\Servers\UnwiredServer\bin
\install-sup-sqlany11.bat install [auto|manual]
cdb_servername cdb_serverhost cdb_serverport
[shared_data_path service_username service_password]
```

When data-tier is on the same host as Unwired Server in a personal or evaluation install

1. On the Unwired server node with data-tier installed, update properties by running:

```
<UnwiredPlatform_InstallDir>\Servers\UnwiredServer\bin
\updateProps.bat -nv "cdb.threadcount=200"
```

   Choose an appropriate server threadcount according to your cluster performance requirements.
2. To change the maximum pool size, edit the `<UnwiredPlatform_InstallDir>\Servers\UnwiredServer\Repository\Instance\com\ybase\djc\sql\DataSource\default.properties` to modify the maxPoolSize property.
3. Stop Unwired Server services.
4. Reconfigure Unwired Platform services by running:

```
<UnwiredPlatform_InstallDir>\Servers\UnwiredServer\bin
\configure-mms.bat <clusterName>
```

Using a Different Database and Log Path

Sybase recommends that you always use the default database and log path location for Unwired Platform databases. However, if you must move these files, you must ensure the servers know where to locate the system data. Otherwise, services cannot start.

By default, when you create a database in SQL Anywhere, the database and log are created in the home directory. If you change this directory, you must supply the correct information every time you restart a data service. To avoid having to supply this information repeatedly, you can use the **dblog** utility to set the correct filename and path.

To move a log from its default location to a new drive, use a command similar to this one:

```
dblog -t D:\databaseLog\default.log -m E:\databaseLog
\default_mirror.log
C:\databases\default.db
```

This example, moves `default.log` from its default location of `C:\databases`, to the `D:\` drive, and sets up a mirrored copy on the `E:\` drive. Sybase recommends naming the log files specified by -t and -m options differently.

You can use this same syntax for other databases, simply by changing the log and database file names.

See also

- *Sample Backup and Recovery Plan* on page 265
- *Initializing a New Consolidated Database* on page 25
- *Changing the Consolidated Database Server Thread Count and Pool Size* on page 26
- *Tuning RBS Synchronization* on page 43

Deploying the Monitoring Database

You can use the installed database or an existing database as the monitoring database. Irrespective of which type of database you deploy, you want to set it up in a way that avoids unduly impacting runtime performance of the system.

Best practices for deploying a monitoring database include:

- Use the Unwired Platform installer to install the monitoring database on a seperate host for improved performance. If you have already installed the monitoring database on the same host as another component, see the corresponding isolation topic that follows.
- If a monitoring database is shared among more than one cluster, size the monitoring server accordingly.
- Choose the correct size of connection pools to the monitoring database from each Unwired Server in accordance with the configuration for the CDB connection pools, which is normally unlimited.
- Put procedures in place to purge data from this monitoring database on a regular basis.
- Create and enable monitoring profiles judiciously or use the monitoring profile named default, which enables monitoring on all packages in all domains.

  Care should be taken to have adequate set up for monitoring so that the system performance is within acceptable boundaries. Excessive monitoring has a detrimental impact to performance of each server in the cluster. Monitoring is performed on separate threads in the server, so monitoring requests do not directly impact Unwired Platform performance. However, the effect of these background threads has an overall impact to the server. For example, in the default monitoring profile, expect a three percent increase to response times and throughput. For information about the features of monitoring, see the *Runtime Monitoring* chapter. For setup steps, see *Creating and Enabling a Monitoring Profile* in Sybase Control Center online help.

See also

- *Status and Performance Monitoring* on page 210
- *Monitoring Database Schema* on page 372
- *Deploying the CDB* on page 25

Setting Up an Existing Database for Monitoring

You can use any SQL database provided that the existing version number of that existing database matches the version required by Unwired Platform.

Setting up the existing database requires some changes to the schema. Once setup, you can configure Unwired Platform to use this database.

1. Use dbisql to run the SQL scripts that set up the monitoring schema. This utility is located in `<UnwiredPlatform_InstallDir>\Servers\SQLAnywhere11\BIN32`.
   - For SQL Anywhere databases:
     - `init-monitoring-tables-asa.sql` – sets up the SQL Anywhere database with a general monitoring schema.
     - `ASA_MONITORING_DDL.sql` – sets up the SQL Anywhere database with a cache monitoring schema.
   - For ASE databases:
     - `init-monitoring-tables-ase.sql` – sets up the Adaptive Server® database with a general monitoring schema.
     - `ASE_MONITORING_DDL.sql` – sets up the Adaptive Server database with a cache monitoring schema.

   By default, these scripts are installed in `<UnwiredPlatform_InstallDir>\Servers\UnwiredServer\config`.
2. Create a new data source for the monitoring database in the **default** domain.
3. In Sybase Control Center, configure Unwired Server to use this database instance. See *Configuring Monitoring Performance Properties* in Sybase Control Center online help.
4. Configure monitoring behavior accordingly.

See also

- *Isolating the Monitoring Database* on page 29
- *Changing the Default Monitoring Port* on page 30

Isolating the Monitoring Database

Install a new data tier node and configure the connection details to it.

1. Perform a data-tier only install on a new data tier node.

   This installs the monitor database as a service.
2. Disable the consolidated and cluster database and advantage database services.

3. Connect to Unwired Server cluster from Sybase Control Center and:
   a) In the navigation pane, expand the **default** domain node, then click **Connections**.
   b) Click the **Connections** tab.
   c) Select monitordb connection, and click Properties.
   d) Update the values to reflect the connection properties of the new host node.
   e) Click Save.

See also

- *Setting Up an Existing Database for Monitoring* on page 29
- *Changing the Default Monitoring Port* on page 30

Changing the Default Monitoring Port

By default, the monitoring database uses port 5600, and the service name is SybaseUnwiredPlatformMonitorDatabase. To change this port, you need to modify `configure-sup.xml`.

1. Stop the monitoring database service.
2. In a text editor, open `configure-sup.xml`.
3. Locate `<target name="monitor_db_options">`, and set a new port in this line:
   `-x tcpip(host=${monitoringdb.serverhost};PORT=5600)`
4. Save the file.
5. Restart the service.

See also

- *Setting Up an Existing Database for Monitoring* on page 29
- *Isolating the Monitoring Database* on page 29

EIS Connections

Prepare Unwired Server to establish connections with different drivers and connectors, especially to non-Sybase data sources like SAP.

Unwired Server includes and uses the appropriate drivers for Sybase databases like Adaptive Server and SQL Anywhere databases. It also has the necessary drivers to connect to Web services.

However, for non-Sybase data sources such as SAP®, you must set up Unwired Server to establish connections to them.

Depending on the type of connection required, certain files need to be copied to Unwired Server directories. Repeat this process on each server in the cluster.

See also

- *EIS Connection Management Overview* on page 162
- *EIS Data Source Connection Properties Reference* on page 308

Preparing Unwired Server to Connect to JDBC Databases

Connections to Oracle, DB2, and SQL Server databases must be prepared for by downloading the appropriate JDBC driver and installing them to the correct location.

Once you have placed drivers in the correct location for Unwired Server, you can connect to JDBC data sources correctly.

1. Shut down Unwired Server.
2. Download the driver.

JDBC driver	URL
Oracle	*http://www.oracle.com/technology/software/tech/java/sqlj_jdbc/index.html*
DB2	*http://www-306.ibm.com/software/data/db2/express/download.html*
SQL Server	*http://msdn.microsoft.com/en-us/data/aa937724.aspx*

3. Place the JDBC driver in the `<UnwiredPlatform_InstallDir>\Servers\UnwiredServer\lib\3rdparty` directory.
4. After copying the driver file, restart Unwired Server.
5. Repeat these steps on each server in the cluster.
6. If Unwired Server instances are installed as windows services, remove then re-install all of the services by using **sup-server-services.bat**.

   For details, see *Unwired Server Service (sup-server-service) Utility*.

Preparing Unwired Server to Connect to SAP using Java Connectors

Unwired Server can connect to SAP data sources with Java Connectors (JCo). With the correct security setup, single singon (SSO) authentication can also be implemented.

Prerequisites

Before you can access the SAP Web site, you must have an SAP account. Sybase also recommends that you make all changes concurrently in your server clusters.

Task

After installing the required SAP files see the *Security* chapter for details on configuring Unwired Server to use SSO.

SAP External Libraries Overview

Understand the purpose of the various external libraries you need to install into Unwired Platform to enable communication with an SAP EIS.

- **SAP Java Connector (JCo) –** required for both Unwired WorkSpace and Unwired Server where connections to an SAP EIS are required. This is not required if you are accessing SAP through a Web service.
- **SAP Cryptographic Libraries –** required by Unwired Platform to enable Secure Network Communications (SNC) between Unwired Server or Unwired Workspace and the SAP EIS.
- **SAP SSO2 Token Libraries –** optional for Unwired Server. These libraries are only required if you use single sign-on (SSO) with SSO2 tokens **and** want to enable persisted token caching. The default, and recommended approach, is to use the Unwired Server authentication timeout interval setting, which does not require these libraries.
- **SAPCAR utility –** required to extract files from certain downloaded SAP files such as the SAP cryptographic library and RFC library.

See also

- *Installing the SAPCAR Utility* on page 33
- *Installing the SAP Java Connector (JCo) and Latest RFC Libraries on Unwired Server* on page 33
- *Installing the SAP Cryptographic Libraries on Unwired Platform* on page 34
- *Installing DOE-C on Unwired Server* on page 36

Downloading and Installing SAP Libraries

The location to which you install your SAP libraries depends on your hardware platform, architecture, and installation type.

When installing SAP libraries, keep in mind that:

- Unwired WorkSpace runs in 32-bit mode, and requires 32-bit libraries whether on a 32 or 64-bit machine.
- Unwired Server can run in either a 32 or 64-bit environment and requires corresponding library types.

For example, if you are running Unwired Server and Unwired WorkSpace on the same 64-bit machine, download and install 64-bit JCo libraries to specified Unwired Server locations and 32-bit libraries to specified Unwired WorkSpace locations.

Installing the SAPCAR Utility

Unzip and install the latest SAPCAR utility on your Unwired Server or Unwired WorkSpace host, which you can use to extract the contents of compressed SAP files, for example RFC and cryptographic library files.

The installation package is available to authorized customers on the SAP Service Marketplace. There are different distribution packages for various hardware processors. Select the package appropriate for your platform.

1. Go to the SAP Web site at *http://service.sap.com/swdc*.
2. From the SAP Download Center, navigate to **Support Packages and Patches** > **Browse our Download Catalog** > **Additional Components**
3. Select **SAPCAR**.
4. Select the current version. For example, **SAPCAR 7.10**, then select and download the SAPCAR appropriate for your platform.

See also

- *SAP External Libraries Overview* on page 32
- *Installing the SAP Java Connector (JCo) and Latest RFC Libraries on Unwired Server* on page 33
- *Installing the SAP Cryptographic Libraries on Unwired Platform* on page 34
- *Installing DOE-C on Unwired Server* on page 36

Installing the SAP Java Connector (JCo) and Latest RFC Libraries on Unwired Server

Unzip and install the contents of the latest SAP Java Connector (JCo) file in all nodes of your Unwired Server cluster. The JCo 2.1.*x* file does not include the latest RFC library, so separately download and install RFC library version 7.10 (non-unicode), replacing the bundled 6.40 version.

Prerequisites

Download and install the SAPCAR utility, which is required to extract the contents of the RFC library download file.

Task

1. Download and install (depending on your machine type) Microsoft Visual C/C++ 2005 Redistributable Service Pack 1 or 64-bit packages.

   Refer to *SAP note 684106* for details and download information.
2. Download the RFC library version 7.10 for your platform and architecture.

Refer to *SAP note 413708* for details and download information. Do not download the new SAP NetWeaver RFC library. Select and download the latest librfc SAR file archive, which contains `librfc32.dll`.

3. Unzip the librfc SAR file archive using the SAPCAR utility.
4. Copy or replace the extracted `librfc32.dll` (32-bit or 64-bit depending on your platform) into:
   - On 32-bit machines copy the 32-bit dll to `C:\WINDOWS\system32`
   - On 64-bit machines copy the 64-bit dll to `C:\WINDOWS\system32`
5. Download the JCo Library:
   a) Go to the SAP Web site at *http://service.sap.com/connectors* and download the latest SAP JCo version 2.1.x for your platform.
   b) Navigate to **SAP Java Connector** > **Tools & Services**.
   c) Select and download the most current SAP JCo 2.1.*x* release, for example SAP JCo Release 2.1.9. Unwired Platform supports the latest 2.1.*x* version of JCo, but does not support JCo versions 3.*x*.
   d) Select the JCo for your particular platform and processor.
6. Copy (or replace, if it already exists) `sapjco.jar` (64-bit or 32-bit depending on your platform) from the JCo Connector package into `<UnwiredPlatform_InstallDir>\Servers\UnwiredServer\lib\3rdparty`.
7. Copy (or replace, if it already exists) `sapjcorfc.dll` (64-bit or 32-bit depending on your platform) from the JCo Connector package into `<UnwiredPlatform_InstallDir>\Servers\UnwiredServer\lib`.

See also

- *SAP External Libraries Overview* on page 32
- *Installing the SAPCAR Utility* on page 33
- *Installing the SAP Cryptographic Libraries on Unwired Platform* on page 34
- *Installing DOE-C on Unwired Server* on page 36

Installing the SAP Cryptographic Libraries on Unwired Platform

Installation and configuration is required if you want to configure Secure Network Communications (SNC) for Unwired Platform SAP JCo connections. SNC may be required by the SAP EIS in question, if SSO2 tokens or X.509 certificates are used for connection authentication.

Prerequisites

Download and install the SAPCAR utility, which is required to extract the contents of the cryptographic library.

Task

Unzip and install the contents of the latest SAP Cryptographic archive on your Unwired Server host. There are different distribution packages for various hardware processors.

Make sure you are installing the correct libraries for your environment, and into folders based on the particular architecture of your machine.

1. Go to the SAP Web site at *http://service.sap.com/swdc* and download the latest SAP cryptographic library suitable for your platform.
   a) Navigate to **Installations and Upgrades** > **Browse our Download Catalog** > **SAP Cryptographic Software** > **SAP Cryptographic Software**.
   b) Select and download the platform specific file.
2. Create a directory in which you unzip the Cryptographic zip file. For example: `C:\sapcryptolib`.
3. Copy the appropriate Windows cryptographic library for your machine (for example, 90000101.SAR) to the `C:\sapcryptolib` directory.
4. Open a command prompt and navigate to `C:\sapcryptolib`.
5. Extract the SAR file. For example:

   SAPCAR_4-20002092.EXE -xvf C:\90000101.SAR -R C:\sapcryptolib
6. Depending on your platform:
   - 64-bit – delete the `nt-x86_64` directory from the `C:\sapcryptolib` directory. Copy the contents of the `nt-x86_64` subdirectory to `C:\sapcryptolib`.
   - 32-bit – delete the `ntia64` and `nt-x86_64` directories from the `C:\sapcryptolib` directory. Copy the contents of the `ntintel` subdirectory to `C:\sapcryptolib`. Delete the `ntintel` subdirectory.
7. Add the SECUDIR environment variable based on machine type and Unwired Platform component:
   - 64-bit Unwired Server – requires access to the 64-bit crypto libraries. Set SECUDIR in `<UnwiredPlatform_InstallDir>\UnwiredPlatform\Servers\UnwiredServer\bin\usersetenv.bat` to point to the location of the 64-bit crypto libraries. For example:
     `set SECUDIR=C:\sapcryptolib`
   - 32-bit Unwired Server – requires access to the 32-bit crypto libraries. Set SECUDIR in `<UnwiredPlatform_InstallDir>\UnwiredPlatform\Servers\UnwiredServer\bin\usersetenv.bat` to point to the location of the 32-bit crypto libraries. For example:
     `set SECUDIR=C:\sapcryptolib`
   - Unwired WorkSpace – runs in 32-bit mode regardless of machine type and requires access to the 32-bit crypto libraries. Set SECUDIR in `C:\Sybase\UnwiredPlatform\Eclipse\UnwiredWorkSpace.bat` above the line SUP_ROOT=<SUP_HOME>. For example:

```
set SECUDIR=C:\sapcryptolib
```

See also

- *CertificateAuthenticationLoginModule Authentication Provider* on page 112
- *Single Sign-on for SAP* on page 123
- *Preparing Your SAP Environment for SSO* on page 128
- *Certificate Authentication Properties* on page 303
- *SAP External Libraries Overview* on page 32
- *Installing the SAPCAR Utility* on page 33
- *Installing the SAP Java Connector (JCo) and Latest RFC Libraries on Unwired Server* on page 33
- *Installing DOE-C on Unwired Server* on page 36

Installing DOE-C on Unwired Server

Install the DOE Connector (DOE-C) on Unwired Server hosts that enable communication between Unwired Server and the SAP system

See the latest version of the *Sybase DOE Connector Installation Guide* for details.

See also

- *SAP External Libraries Overview* on page 32
- *Installing the SAPCAR Utility* on page 33
- *Installing the SAP Java Connector (JCo) and Latest RFC Libraries on Unwired Server* on page 33
- *Installing the SAP Cryptographic Libraries on Unwired Platform* on page 34

Changing Connections to Production Data Sources

Platform and domain administrators can change endpoint connection information when moving applications from development or test environments to production environments.

Review this list to determine what connection elements may be affected when transitioning between these different environments:

1. You can change endpoint connection mapping when the mobile business objects (MBOs) are deployed to the production Unwired Server.

   Make these changes using either Unwired WorkSpace development tools for design-time changes, or Sybase Control Center for Unwired Platform for deployment-time changes.
2. You need not change MBOs, if the developer uses production data source connection credentials. MBOs that use user name and password personalization keys to authenticate server connections, do not use the user name and password specified in the server connection.

The client user credentials are used to establish connection with the back-end. Administrators should determine from their development team which MBOs are affected. You must still remap connections to production values.

3. (Optional) Tune properties (such as connection pool size), to improve the performance of production-ready applications.

Device User Credentials and EIS Connections

Administrators should be aware that developers may map a connection's user name and password properties to system-defined "username" and "password" personalization keys, respectively. This creates a new connection for each client and the connection is established for each request (no connection pooling).

During deployment of this kind of MBO package, administrators must map the design-time connections to the production EIS server connections. However, the username and password for the selected server connection are replaced by the user name and password propagated from the device application via the personalization key.

Synchronization Model Performance Tuning

Whether you use the replication-based synchronization model or the message-based one, tuning is a function of providing the highest throughput while limiting CPU and memory consumption to reasonable operational levels. You can tune performance immediately after installing Unwired Platform components with all runtime artifacts running and stable. You can refine it as part of regular system administration duties in an ongoing capacity, or when new runtime components are deployed.

Tuning recommendations vary depending on your model.

See also

- *Applying Performance Tuning Changes if Unwired Server is a Service* on page 150
- *Changing the Consolidated Database Server Thread Count and Pool Size* on page 26
- *Configuring the IIOP Socket Listener* on page 143
- *Configuring Unwired Server Performance Properties* on page 149

Performance Considerations for RBS Synchronization

Understand fundamentals of replication-based (RBS) synchronization so you can understand how to improve performance for this synchronization model.

Replication-based synchronization has two distinct operation and data transfer phases where only differences are transferred: upload (where only updates from the mobile client to the server are integrated with data in the mobile middleware cache and pushed to the EIS) and download (where EIS delta changes are determined for, and supplied to, the specific client). These phases are conducted as discrete transactions to protect the integrity of synchronization. Replication-based synchronizations are carried out within a single data transfer session,

allowing for exchanges of large payloads. Therefore the goal of tuning performance for RBS is to ensure those transfers of large volumes are handled effectively and not to create bottlenecks in the runtime.

Considerations that affect performance can be seperated into synchronization phases and architecture components.

Replication-based synchronization use cases center around three primary phases in moving data between server and client, each of which need to be considered when choosing performance setting values. Each phase describes mobility environment components, and how they affect performance: MBO development and data design, EIS data models and runtime servers, and Unwired Server runtimes. These phases are:

- Initial synchronization – where data is moved from the back-end enterprise system, through the mobile middleware, and finally onto the device storage. This phase represents the point where a new device is put into service or data is reinitialized, and therefore represents the largest movement of data of all the scenarios. In these performance test scenarios, the device-side file may be purged to represent a fresh synchronization, or preserved to represent specific cache refreshes on the server.
  For this phase, the most important performance consideration is the data and how it's partitioned (EIS design), and loaded by the MBO (MBO development) using operation parameters, synchronization filter parameters). Synchronization parameters determine what data is relevant to a device; they are used as filters within the server-side cache. Load parameters determine what data to load from the EIS into the Unwired Platform cache.
- Incremental synchronization – involves create, update, and delete (CUD) operations on the device where some data is already populated on the device and changes are made to that data which then need to be reconciled with the middleware and the back-end enterprise system. When create and update operations occur, changes may be pushed through the middleware cache to the back end, reads may occur to properly represent the system of record, for example, data on the back end may be reformatted, or both. This scenario represents incremental delta changes to and from the system of record.
  As in the initial synchronization phase, the EIS accounts for the bulk of the device synchronization response time: a slow EIS consumes resources in the Unwired Server, with the potential to further impede devices that are competing for resources in connection pools. Additionaly, the complexity of the mobile model, measured by the number of relationships between MBOs, has a significant impact on create, update, and delete operation performance. Shared partitions among users or complex locking scenarios involving EIS operations can become a major performance concern during device update operations. Cache and EIS updates are accomplished within the scope of a single transaction, so other users in the same partition are locked out during an update of that partition. Consider denormalizing the model if complex relationships cause performance concerns.
- Data change notification (DCN) – changes to the back-end data are pushed to the mobile middleware and then reconciled with the device data. DCN is typically observed in the context of additional changes to the device data so that changes from both device and back end are simultaneously impacting the mobile middleware cache.

DCN efficiently updates the Unwired Server because it does not require the Unwired Server to poll the EIS or to refresh the cache based on a schedule. EIS DCN application to the cache is independent of the client synchronizations. If DCN data is located in a shared partition, multiple devices benefit from the single EIS update to the cache. There are several ways to materially improve DCN performance:

- Use a load-balancer between the EIS and the Unwired Server – DCNs can be efficiently applied across an Unwired Platform cluster, as each node in the cluster helps to parse incoming payloads.
- Combine multiple updates into a single batch.
- Run DCNs from a multithreaded source to parallelize updates. Note that there is a diminishing return beyond three to four clients, in large part due to the nature of the model.

Different models exhibit different performance characteristics when applying updates, so proper analysis of application behavior is important.

See also

- *Performance Considerations for MBS Synchronization* on page 45

Overview of RBS Tuning Recommendations

RBS tuning recommendations are designed to maximize bandwidth between the relay server and its outbound enablers, Unwired Server, and the EIS connections.

Sybase recommendations touch on components outside and inside the LAN:

- For large RBS payloads greater than 4MB, increase the bandwidth between the relay server and the Unwired Platform cluster nodes by increasing the number of relay server outbound enablers (RSOEs) and increasing the shared memory of the relay servers.
- Ensure adequate processing bandwidth for peak conditions by setting an adequate replication thread count for the entire cluster (the combination of each node's replication thread count) and the JVM memory settings. The replication thread count is the point where Unwired Server and consolidated database (CDB) throttling, or limiting, occurs.

  The speed of the consolidated database (CDB) storage and storage controllers is the single most important factor in providing good system performance. Staging of mobile data is performed within the CDB in a persistent manner such that if a device synchronizes. The CDB database server itself is largely self-tuning, although typical database maintenance is essential for proper performance.
- Provide as many resources as are available as you work back to the EIS. Do not limit connection pools, unless there is a requirement to do so relating to the storage medium.

Relay Server/RSOE Performance Tuning Reference

Use this table to quickly ascertain which relay server and RSOE properties you can adjust to tune performance.

Table 2. Recommendations Generally Suitable for 64-bit Production Servers

Component	Function	Default	Production Recommendation
Relay Server shared memory	Settings for shared memory buffer. Remember that shared memory must account for concurrent connections, especially with large payloads.	10MB	2GB
RSOE deployment	With SQL Anywhere® 11, each RSOE provides an upload and download channel to the relay server.	1 RSOE per server	3 RSOEs for Unwired Server nodes, especially when initial syncs are greater than 4MB

Unwired Server RBS Performance Tuning Reference

Use this table to quickly ascertain which runtime properties you can adjust to tune RBS performance.

Table 3. Recommendations Generally Suitable for 64-bit Production Servers

Property	Function	Default	Production Recommendation
Replication thread count	Controls the concurrent number of threads that service devices for synchronization. This setting controls the amount of CPU used by the Unwired Platform and CDB tiers. If the processor of either the CDB or the Unwired Server is excessively high, you can use this setting to throttle the number of requests and limit contention for resources. Low settings decrease parallelism; high settings may cause undue contention for resources.	10 per server	For a 2-node cluster, use 12 for each node. For a single node, use 24.

Property	Function	Default	Production Recommendation
Synchronization cache size	The maximum amount of memory the server uses for holding table data related to device users, network buffers, cached download data, and other structures used for synchronization. When the server has more data than can be held in this memory pool, the data is stored on disk.	50MB	1GB
JVM minimum heap size	The minimum memory allocated to the differencing and cache management functions of the server.	64MB	2GB
JVM maximum heap size	The maximum memory allocated to the differencing and cache management functions of the server.	256MB	4GB

CDB Performance Tuning Reference

Use this table to quickly ascertain which CDB properties you can adjust to tune performance.

Because the RBS model uses staging and replication technology, as well as using a differencing algorithm to determine what to synchronize to each device, the CDB is one of the most critical components for performance in terms of processing power, memory, and disk performance. The CDB must also be scaled vertically (on larger hardware) because it supports all of the nodes of the cluster.

Table 4. Recommendations Generally Suitable for 64-bit Production Servers

Property	Function	Default	Production Recommendation
CDB thread count	Sets the maximum multiprogramming level of the database server. It limits the number of tasks (both user and system requests) that the database server can execute concurrently. If the database server receives an additional request while at this limit, the new request must wait until an executing task completes.	20	200

Property	Function	Default	Production Recommendation
CDB initial cache size	Sets the initial memory reserved for caching database pages and other server information. The more cache memory that can be given the server, the better its performance.	24MB	8GB
CDB disk configuration	Isolate the data and log on distinct physical disks. Of the two files, the log receives the most activity. Use disk controllers and SAN infrastructure with write-ahead caching in addition to high-speed disk spindles or static memory disks. The same database must support all cluster members. The faster these database drives perform, the better the performance on the entire cluster.	1 disk	2 disks 10K RPM
Default (CDB) connection pool maximum size	Sets the maximum (JDBC) connection pool size for connecting to the CDB from each cluster member. In the example production configuration, each Unwired Server is allowed more connections than the number of threads set in the CDB thread configuration. This ratio helps to ensure that the CDB database server is the control point for limiting resource contention in the cluster. In a server configured with the CDB connection pool settings set to 0, the actual number of connections used will correspond to the number of Unwired Server threads in use. The number of internal connections in use at any one time per thread varies, although fewer than four is typical.	100	0 (unlimited)

EIS Connection Performance Tuning Reference

Use this topic to quickly ascertain which EIS connectivity properties you can adjust to tune performance.

Property	Function	Default	Production Recommendation
EIS connection pool maximum size	Ensure that adequate EIS resources are allocated for servicing the mobile infrastructure. The actual number of connections necessary varies, based on the maximum number of Unwired Platform threads in use at any time and the duration it takes for the EIS to respond. If possible, allocate a connection for each thread (or leave the setting unbounded). If you must limit the number of EIS connections in the pool to a lower number than the number of Unwired Platform threads and you experience timeouts, you may need to adjust the EIS connection timeout values ; however, these connections will impede other threads competing for EIS connections.	10	0 (unlimited)

Tuning RBS Synchronization

Tune your production environment after all components have been deployed and started.

In a production environment, system components are installed on distinct runtime tiers— one for the relay server farm, one for the Unwired Server cluster, one for the runtime data (that is, the CDB and the cluster database, and the monitoring database).

1. Isolate the monitoring database from the CDB server.

   See *Isolating and Setting Up Production Monitoring Databases*.
2. Disable all enabled monitoring profiles that currently exist.

   Normally, monitoring in a cluster configuration occurs at two levels — the cluster and the domain. Sybase recommends that you turn off monitoring when assessing performance.

   a) In navigation pane of Sybase Control Center, click **Monitoring**.
   b) In the administration pane, select the profile name and click **Disable**.
   c) Validate that all monitoring is disabled by opening `<UnwiredPlatform_InstallDir>\Servers\UnwiredServer\Repository\Instance\com\sybase\sup\server\monitoring\MonitoringConfiguration\domainLogging.properties`, and verifying that “status”, “enabled”, and “autoStart” are set to false.

See *Monitoring Profiles* in the Sybase Control Center online help.

3. In Sybase Control Center, stop all Unwired Servers in all clusters.
4. Stop the CDB service on the data tier node.
5. Use the dblog utility to isolate the CDB disk and log on to fast storage.

   For details, see *Using a Different Database and Log Path*.
6. Set the new CDB thread count, pool size, and initial cache size.

   See *Changing the Consolidated Database Server Thread Count and Pool Size*.
7. Restart the CDB service and all Unwired Servers.
8. In Sybase Control Center, adjust the EIS connection pools to high values:
   a) In navigation pane of Sybase Control Center, click **Connections**.
   b) In the administration pane, set the Max Pool Size property for each EIS connection type.

      If the Max Pool Size property does not appear for a connection, you must manually add it.
9. In Sybase Control Center, update the Unwired Server RBS synchronization properties on the primary server.
   a) In the left navigation pane, click **Servers**>**SeverName**>**Server Configuration**.
   b) In the right administration pane, click the **Replication** tab.
   c) Adjust the synchronization cache size, and the replication thread count for the entire cluster.
   d) Repeat for all servers, so that each node has an equal thread count.
10. In Sybase Control Center, set thread stack size and JVM heap sizes for the server.
    a) In the left navigation pane, click **Servers**>**SeverName**>**Server Configuration**.
    b) In the right administration pane, click the **General** tab then click **Performance Configuration**.
    c) Adjust the thread stack size and JVM heap sizes to the new values.
    d) Repeat for all servers.
11. Restart all Unwired Servers.

Testing CPU Loads for RBS

Perform a mixed-load test to ascertain whether the throughput to the CDB and EIS is sufficient. Do this by testing then observing CPU load trends.

1. Run a typical application maximum mixed-load test on the Unwired Server (5% initial sync, 95% subsequent sync) and observe the CDB CPU load.

   Do not run this on the relay server; you want to first ascertain the maximum throughput of the Unwired Platform and CDB tiers.
2. If the CDB CPU load is too high, too many threads could be impacting performance:
   a) Decrease the replication thread count on each server in Sybase Control Center.

b) Restart all servers.
c) Observe the CDB CPU load again.

3. If the CDB CPU load is low, fine tune the replication thread count until one yields the maximum through put. Check this by:
   a) Note the client response times of other synchronization types and try increasing the replication thread count in small increments.

      Other types of synchronization include initial to initial, subsequent to subsequent, similar payload, cache policy, and so on.
   b) Restart the all servers.
   c) Repeat this process until the synchronization response times are as low as possible for the same number of clients for all types of synchronizations.

      Once the relative client response time is as low as possible without further increasing the replication thread count, you have reached the configuration that yields the maximum throughput. In other words, tune the thread count to yield the best overall response times. Usually, this thread count number is small.
4. Repeat the mixed-load test on EIS backends.

   This checks for EIS latencies and cache policies, both of which can change the performance of the server cluster.
5. Introduce relay server into the environment, and repeat the mixed-load test.
6. Once the environment has stabilized and tuned, enable on and configure only the monitoring profiles you need.

Performance Considerations for MBS Synchronization

Understand fundamentals of message-based (MBS) synchronization so you can understand how to improve performance for this synchronization model.

MBS is conducted via the reliable exchange of messages, trading off between efficiency and immediacy. Some messages arrive earlier than in replication-based synchronization because the payload is spread out over many relatively small messages that are delivered individually, as opposed to being delivered as a single large download. Therefore the goal of tuning performance for MBS is to multiple transfers of small volumes are handled quickly to keep changes in the correct order.

MBS use cases center around three primary phases in moving data between server and client, each of which need to be considered when choosing performance setting values. Each phase describes mobility environment components, and how they affect performance: MBO development and data design, EIS data models and runtime servers, and Unwired Server runtimes. These phases are:

- Initial subscription – A mobile application subscribes to an MBS package to receive data as “import” messages from the server. Upon the receipt of the subscription request from the device, the server checks security and executes a query against the consolidated database (CDB) to retrieve the data set, which it then turns into a series of import messages to be sent to the device. The maximum message size is currently fixed at 20KB. Increasing

the maximum allowable size would allow the same amount of import data with fewer messages. However, in some devices, large message size can trigger resource exhaustion and failures.

The subscription phase is the most expensive or time consuming portion of the life cycle, especially for a large initial data set. However, compared to RBS, MBS can take longer to populate the data on the device database because of the aggregation of fine-grained messages. This latency is due to the additional cost of providing reliability to the delivery of every message. In addition, a significant amount of processing is incurred on both the server and the client side to marshal messages. While this marshaling may not necessarily impact the server, it places a heavy burden on the device, especially if the device is on the low end of the performance spectrum. The message import is also limited by the device's nonvolatile storage write performance.

- Subsequent Synchronization – Device-side data changes due to the user interacting with a mobile application. It is important that you understand the unit of change in MBS. An operation with an associated tree of objects with a containment (or composite) relationship is considered a unit of change. Changes are wrapped in a message with the appropriate operation type: create or update. (The delete operation requires only the primary key that identifies the root of the tree to be transmitted, rather than the entire object tree). Upon receipt of the message, the server replays the operation against the EIS and returns a replay result message with one or more import messages that reflect the new state of the data. Unlike RBS, the unit of change is pushed to the device as soon as the changes are ready, so subsequent synchronization is occurring in the form of many messages. As a result, the synchronization happens over time as a stream of messages.

  The subsequent synchronization phase addresses mobile devices sending CUD requests to Unwired Platform and receiving responses and updates as a result of the operations. In general, the limiting factor for performance is on the EIS as the CUD requests are replayed. You may experience Unwired Server performance issues if requests are spread over a number of back-end systems. If the EIS response time is large, you may need to allocate additional threads to replay requests concurrently against the EIS.
- Incremental Synchronization – The updates sent to the device as messages due to DCN from the EIS. DCN is the most efficient way for a back-end system to notify Unwired Platform of changes to data mobilized to the device (as opposed to polling for EIS changes).Because MBS uses push synchronization to send out the updates as import messages to the devices, each DCN normally causes updates to device data over the MBS channels, based on a configurable schedule within the server. The main reason for this behavior is the concern for activity storms arising from batched DCNs where many granular changes on the cache cause flurries of messages that might be better consolidated first on the server. Currently, most EIS backends are not event-driven and DCNs are batched. Hence, having batched triggers directly driving an event-based model can decrease efficiency without increasing data freshness.

See also

- *Performance Considerations for RBS Synchronization* on page 37

Overview of MBS Tuning Recommendations

MBS tuning recommendations are designed to maximize throughput of messages, as there are unique implications to using messages as the medium of data exchange. The MBS processing limit (throttle) is determined by the number of inbound message queues. Any Unwired Server within a cluster can process messages on inbound queues. The number of inbound queues is equal to the number of parallel MBS package requests that the cluster can concurrently service.

Sybase recommendations touch on components outside and inside the LAN:

- Add multiple server nodes to increase data marshaling. Testing has indicated that one Unwired Server node is capable of sustaining around 70 messages per second to devices. A second Unwired Server node provides an additional 60% increase in delivery capability. This increase is because the second MBS node increases message marshaling and transmission capability for the entire device population.
- Device performance and number of devices deploy. Device processing capacity limits the capacity of each message to 20KB. Due to serialization of data to JSON, a roughly 2X increase in size is observed within our data model, i.e. 4MB worth of data is represented by 9MB of serialized information for packaging into messages. As a result, the number of import messages is in the low 400s. The 2X factor is only an estimate for a moderately complex application; the true cost depends on the number of attributes (and the datatypes) of each MBO in the model. Furthermore, if push messages are to be generated for a device, you should tune the environment after executing download SQL queries against the cache: the cost of the query depends on the complexity of the download logic (number of MBO relationships) in addition to synchronization timestamp comparison. For a large number of devices, this cost can be considerable.
- Approach initial synchronization carefully, and follow a sound rollout policy for new MBS mobile applications. The easiest approach is to spread the rollout over a period a time to avoid a large number of devices attempting to download large amounts of data. Excessive load not only slows the rollout, but may also impact performance for existing MBS applications. Apart from Unwired Platform capacity and connectivity bandwidth, the actual time required to perform the initial synchronization depends mostly on the capability of the device.
- Allocate additional threads to replay requests concurrently against the EIS. Choosing a value depends upon:
  - The number of concurrent CUD operations that an EIS can handle without causing degradation
  - The number of simultaneously active EIS operations the load is spread across
  - The average response time for various EIS operations
  - Any lock contention points in the mobile model that exist due to shared data partitions
  - The total application landscape (Unwired Platform does not assign threads on a per package basis. All threads are available to service CUD operations for any of the deployed MBS packages.)

- The Unwired Server data change check frequency for each package. The higher the check frequency, the higher the cost, but data freshness may not actually increase. The gating factor is the frequency of updates from the EIS. If the cache is configured to refresh at midnight, after the EIS performs certain batch processing, it is most efficient to configure the scheduler to check for changes sometime after the cache refresh is completed. For an EIS that uses DCN to update the cache, the amount and frequency of changes depends on the business process. Understanding the data flow pattern from EIS to cache is crucial to determine how frequently the Unwired Server is to check for changes to be pushed to the devices.
- Batch multiple DCN updates in a single notification to reduce overhead. However, a batch that is too large may impact device synchronization response due to contention. A storm of DCNs can create significant work for devices. The tradeoff is efficiency versus performance (responsiveness).
- During the packaging of data into messages, serialization consumes a significant amount of memory. Avoid garbage collection activities, especially during MBS application rollout, to prevent "choking."

Relay Server/RSOE Performance Tuning Reference

Use this table to quickly ascertain which relay server and RSOE properties you can adjust to tune performance.

Table 5. Recommendations Generally Suitable for 64-bit Production Servers

Component	Function	Default	Production Recommendation
Relay Server shared memory	Settings for shared memory buffer. Remember that shared memory must account for concurrent connections, especially with large payloads.	10MB	2GB
RSOE deployment	With SQL Anywhere® 11, each RSOE provides an upload and download channel to the relay server.	1 RSOE per server	3 RSOEs for Unwired Server nodes, especially when initial syncs are greater than 4MB

Unwired Server MBS Performance Tuning Reference

Use this table to quickly ascertain which runtime properties you can adjust to tune MBS performance.

The majority of MBS tuning revolves around Unwired Server configuration:

Table 6. Recommendations Generally Suitable for 64-bit Production Servers

Property	Function	Default	Production Recommendation
Number of inbound queue*	The concurrent number of threads that service MBS package(s) requests from devices. If the processor of either the messaging database or the Unwired Server is excessively high, you can throttle the number of requests and limit contention for resources using this setting. Low settings decrease parallelism; high settings may cause undue contention for resources.	5 per cluster	100 per cluster in a 2-node cluster
Number of outbound queue*	The number of outbound queues between Unwired Platform and Messaging Services. Messages from these queues are eventually persisted, by the JmsBridge component into a per-device queue belonging to the Messaging Services. Subscription requests can generate a large number of outbound messages and temporarily cause backups in the queues. Since multiple devices can share the same output queue, larger number of queues can reduce delay getting the messages to the intended devices.	25 per cluster	100 per cluster in a 2-node cluster
Check interval for MBS package	Specifies whether the interval system should check for an MBS package to see if there are changes to be pushed to the devices. Align this setting when the cache is being refreshed or updated. If the cache is refreshed every 4 hours, the check interval should also be 4 hours. However, if the cache is only updated via DCN, align the update frequency with the DCN interval accordingly. The check/push generation algorithm is scheduled to be enhanced in future versions.	10 minutes	10 minutes

Property	Function	Default	Production Recommendation
Number of push processing queues*	Number of queues (threads) that execute the change detection function (download SQL execution) to determine if there are changes to be pushed to the device. The number of queues determines the maximum concurrency for change detection. All MBS packages shared the same set of push processing queues.	25 per cluster	25 per cluster in a 2-node cluster.
JVM minimum heap size	The minimum memory allocated to the differencing and cache management functions of the server.	512MB	2GB
JVM maximum heap size	The maximum memory allocated to the differencing and cache management functions of the server.	2GB	6GB

* Cluster-affecting property

Tuning MBS Synchronization

Tune your production environment after all components have been deployed and started.

In a production environment, system components are installed on distinct runtime tiers— one for the relay server farm, one for the Unwired Server cluster, one for the runtime data (that is, the CDB and the cluster database, and the monitoring database).

1. Isolate the monitoring database from the CDB server.

   See *Isolating and Setting Up Production Monitoring Databases*.
2. Disable all enabled monitoring profiles that currently exist.

   Normally, monitoring in a cluster configuration occurs at two levels — the cluster and the domain. Sybase recommends that you turn off monitoring when assessing performance.

   a) In navigation pane of Sybase Control Center, click **Monitoring**.
   b) In the administration pane, select the profile name and click **Disable**.
   c) Validate that all monitoring is disabled by opening `<UnwiredPlatform_InstallDir>\Servers\UnwiredServer\Repository\Instance\com\sybase\sup\server\monitoring\MonitoringConfiguration\domainLogging.properties`, and verifying that “status”, “enabled”, and “autoStart” are set to false.

   See *Monitoring Profiles* in the Sybase Control Center online help.
3. Stop all Unwired Servers in your cluster(s).

4. In Sybase Control Center, change the number of outbound and inbound queues to 100 each for each Unwired Server.
   a) In the left navigation pane, click **Servers**>**SeverName**>**Server Configuration**.
   b) In the right administration pane, click the **Messaging** tab.
   c) Change the default values for inbound and outbound queues.
   d) Repeat for all servers in all clusters.
5. Restart all servers in your cluster(s).
6. In Sybase Control Center, deploy an test MBS package with a representative amount of data for initial subscription.

   For details, see *Deploying and Managing MBO Packages*.

   For example, if the use case dictates that during an initial subscription, the mobile application is to receive 2MB of data, develop the test package to reflect that fact.
7. Start testing by using expected number of devices to perform an initial subscription, and determine if the time to get the initial data set is satisfactory for all devices.

   The maximum messaging throughput 70 messages per second in a wired environment.
   - If the calculated throughput for the test is below this number, it is likely that the connection method (as opposed to the server environment) is the limiting factor. In this case, more devices can be supported without any degradation in server performance.
   - If the test reaches the maximum throughput, the number of devices performing the initial subscription is the maximum one server can handle. Add another server to the cluster for additional message processing power (up to a 60% increase).

CHAPTER 5 Relay Server Clusters

A relay server manages connections to Unwired Servers from the DMZ in which the relay server is installed. When a relay server manages multiple Unwired Server farms, it forms a relay server cluster.

A relay server cluster consists of:

- One or more relay servers running in the corporate DMZ.
- (Optional) A load balancer to direct requests from the mobile devices to a group of relay servers. Multiple relay servers can manage a single farm, however Sybase recommends a load balancer in order to better distribute load among multiple relay servers. A load balancer is not included with Unwired Platform.
- Back-end Unwired Servers running in a corporate LAN. Use a relay server with an Afaria farm that you have also installed either with an earlier version of Unwired Platform or as a standalone version of Afaria that you installed separately.

  Servers that belong to the same cluster are considered to be in the same relay server farm. A client that makes a request through the relay server must specify the server farm using an ID assigned by the platform administrator.
- One or more relay server outbound enabler (RSOE) per back-end server node. Sybase recommends three RSOEs per synchronization type, per Unwired Server.

  The RSOE manages all communication between a server and the relay server farm. Its primary function is to initiate an outbound connection to all relay servers in the farm on behalf of the server.
- Certificates, to encrypt communications for increased security. See *Encrypting Relay Server Communications*.

When all these components are deployed (installed, configured, and started as required), the relay server cluster setup is complete. Over time, you can scale the cluster as required by adding or removing one or more of these components.

See also

- *Unwired Server Clusters* on page 10
- *Implementing an N+2-node Cluster for Optimal Redundancy* on page 14
- *Configuring Push Notifications for the BlackBerry Enterprise Server* on page 175

Relay Server Documentation

Use the Relay Server documentation to obtain additional reference details about the relay server and the RSOE.

This guide provides all Unwired Platform specific implementation details to administrators who want to install and configure relay server. However, for additional reference details, read the *Relay Server User Guide*. Unwired Platform information takes precedence over this document.

If you are viewing this guide as a PDF, you can locate the referenced document at *http://infocenter.sybase.com/*. Search for *Relay Server Documentation* to launch this PDF.

Creating a New Relay Server Cluster

Once you have deployed your Unwired Platform runtime components, you can create one or more relay server clusters, optionally with a load balancer.

Prerequisites

Review *Relay Server Requirements* documented in the *Install Guide*.

Task

A relay server cluster typically is used by a production environment using installed relay server binaries. If you plan to have multiple relay servers deployed to the DMZ, Sybase encourages you to use a load balancer. If you are testing the functionality of an application in a development or test environment, you can use Sybase Hosted Relay Service instead.

1. *Installing Relay Server to Multiple Web Server Host Nodes*

   Use the corresponding relay server binary archive to extract files belonging to a specific host and Web server type.
2. *Configuring a Multinode Relay Server Cluster with Sybase Control Center*

   A multinode relay server cluster includes one or more relay server nodes and multiple RSOEs deployed to corresponding Unwired Server nodes. Use Sybase Control Center to make a multinode relay server cluster operational
3. *Enabling Application Request Routing for IIS Relay Server Hosts*

   Sybase recommends using IIS Application Request Routing (ARR) to balance request loads among IIS instances that host relay servers.

Installing Relay Server to Multiple Web Server Host Nodes

Use the corresponding relay server binary archive to extract files belonging to a specific host and Web server type.

Perform these tasks one or more times as required by your network design.

See also

- *Configuring a Multinode Relay Server Cluster with Sybase Control Center* on page 67

Deploying Relay Server on IIS 7.x Web Servers

Use Microsoft IIS 7.x Web server to host relay server on any supported Windows OS host computers. IIS 7.x is the required version of IIS, if you intend to also implement load balancing with Web Platform Installer.

1. *Installing the Relay Server Components to IIS 7.x on Windows*

   After installing Sybase Unwired Platform, install the relay server component files to Microsoft Internet Information Service (IIS) 7.0 or 7.5 on Windows 2008 or Windows 2008 R2.
2. *Configuring Relay Server to Run as a Windows Service*

   If you installed relay server on an IIS host, Sybase recommends that you set up the relay server as a Windows service to ensure that starts with the host.
3. *Tuning IIS for Improved Relay Server Performance*

   If relay server is installed to an IIS server, Sybase recommends tuning IIS to improve relay server performance. One best practice is to increase the number of connection pools used beyond just a single pool.

See also

- *Deploying Relay Server on IIS 6.x Web Servers* on page 59
- *Deploying Relay Server on Apache Web Servers* on page 62
- *Enabling Application Request Routing for IIS Relay Server Hosts* on page 69

Installing the Relay Server Components to IIS 7.x on Windows

After installing Sybase Unwired Platform, install the relay server component files to Microsoft Internet Information Service (IIS) 7.0 or 7.5 on Windows 2008 or Windows 2008 R2.

Prerequisites

Make sure Microsoft IIS 7.0 or 7.5 is installed on Windows 2008 or Windows 2008 R2, with the Microsoft IIS ISAPI Extensions feature on your server. This feature is not installed by default.

Task

1. Install IIS 7.x server, with ISAPI and GCI extensions.

   These extensions are not installed by default.
2. From the Installation package, get the relevant Relay Server ZIP file:
   - for 32-bit operating systems, use `relayserver.zip`
   - for 64-bit operating systems, use `relayserver_x64.zip`

   The ZIP file contains the required executables and DLLs for the Relay Server.
3. Extract the ZIP file under `C:\Inetpub\wwwroot\` to create the folder structure:

   ```
   C:\inetpub\wwwroot\ias_relay_server
   C:\inetpub\wwwroot\ias_relay_server\Server
   C:\inetpub\wwwroot\ias_relay_server\Client
   ```
4. Verify that all the `DLL` files copied under these folders are of the required version, for example, 11.0.1.2527.
5. Check the Relay Server logs to make sure the first line reflects the new version. For example, from `C:\windows\System32\inetsrv\ias_relay_server_host.log`.

   ```
   I. 2010-12-04 07:22:28. < 3160.- > iAnywhere Relay Server Version
   11.0.1.2527
   I. 2010-12-04 07:22:28. < 3160.- >
   I. 2010-12-04 07:22:28. < 3160.- > Copyright © 2001-2009,
   iAnywhere Solutions, Inc.
   I. 2010-12-04 07:22:28. < 3160.- > Portions copyright © 1988-2009,
   Sybase, Inc. All rights reserved.
   I. 2010-12-04 07:22:28. < 3160.- > Use of this software is
   governed by the Sybase License Agreement. Refer to http://
   www.sybase.com/softwarelicenses
   ...
   ```
6. Modify the System Path variable to refer to include `C:\inetpub\wwwroot\ias_relay_server\Server`.
7. Back up `%SystemDrive%\Windows\System32\inetsrv\config\applicationHost.config`.
8. Open `applicationHost.config` in an editor, and merge the following sections into their respective locations in the file, then save the changes (detailed info follows):
   a) To add an application pool for the Relay Server, edit the `applicationHost.config` file to add the following code to the `<system.applicationHost> » <applicationPools>` section:

   ```
         <add name="RelayServer" queueLength="65535"
   autoStart="true"
   managedRuntimeVersion=""
   managedPipelineMode="Integrated">
             <processModel identityType="LocalSystem"
   idleTimeout="00:00:00" maxProcesses="20"
   pingingEnabled="false"
   ```

```
pingInterval="00:00:30" pingResponseTime="00:01:30" />
          <recycling disallowOverlappingRotation="true">
              <periodicRestart time="00:00:00">
                  <schedule>
                      <clear />
                  </schedule>
              </periodicRestart>
          </recycling>
          <failure rapidFailProtection="false" />
          <cpu resetInterval="00:00:00" />
      </add>
```

b) To add the Relay Server application to the default site, edit the `applicationHost.config` file to add the following code to the `<system.applicationHost> » <applicationPools> » <sites> » <site name="Default Web Site">` section:

```
      <application path="/ias_relay_Server"
applicationPool="RelayServer">
          <virtualDirectory path="/" physicalPath="c:\inetpub
\wwwroot\ias_relay_server" />
      </application>
```

c) To add the Relay Server ISAPI extensions, edit the `applicationHost.config` file to add the following code to the `<system.webServer> » <security> » <isapiCgiRestriction>` section.

```
      <add path="c:\inetpub\wwwroot\ias_relay_server\Client
\rs_client.dll" allowed="true" />
      <add path="c:\inetpub\wwwroot\ias_relay_server\Server
\rs_server.dll" allowed="true" />
```

d) To add the Relay Server handlers, edit the `applicationHost.config` file to add the following code to the `<configuration>` section.

```
    <location path="Default Web Site/ias_relay_server/client">
        <system.webServer>
            <handlers accessPolicy="Execute, Script">
            </handlers>
        </system.webServer>
    </location>

    <location path="Default Web Site/ias_relay_server/server">
        <system.webServer>
            <handlers accessPolicy="Execute, Script">
            </handlers>
        </system.webServer>
    </location>

    <location path="Default Web Site/ias_relay_server">
        <system.webServer>
            <security>
                <authentication>
                    <anonymousAuthentication userName="" />
                </authentication>
                <requestFiltering>
```

```
                    <requestLimits
maxAllowedContentLength="2147483647" />
                </requestFiltering>
            </security>
        </system.webServer>
    </location>
```

9. Open a Web browser, and ensure `http://localhost:80` loads the default page correctly.

   Note: The Relay Server is set up for anonymous access based on these instructions. Proper security needs to be configured for IIS and the Relay Server based on your business requirements.

10. Create a Relay Server service, as described *Configuring a Relay Server to Run as a Windows Service*.
11. Start the Relay Server service, for example:

    C:\inetpub\wwwroot\ias_relay_server\Server > dbsvc -u SUPRelayServer

12. Check Relay Server and IIS logs for errors:
    - Relay Server logs:
      - `C:\windows\System32\inetsrv\ias_relay_server_host.log`.
      - Log specified in the service's `-o` switch, for example: `C:\Sybase\logs\rs.log`.
    - IIS logs:
      - `C:\inetpub\logs\LogFiles\W3SVC1`
      - `C:\Windows\System32\LogFiles\HTTPERR`

Configuring Relay Server to Run as a Windows Service

If you installed relay server on an IIS host, Sybase recommends that you set up the relay server as a Windows service to ensure that starts with the host.

1. At the command line, enter the following command, substituting all parameter values to match your configuration:

   ```
   dbsvc -as -s auto -w SUPRelayServer "C:\inetpub\wwwroot
   \ias_relay_server\server\rshost.exe" -q -qc -f c:\inetpub
   \wwwroot\ias_relay_server\server\rs.config -o c:\Sybase
   \logs\rs.log
   ```

   This command configures the relay server host process (rshost.exe) as a Windows service. By default, the **dbsvc** utility is installed in `<UnwiredPlatform_InstallDir>\Servers\SQLAnywhere11\BIN32\`.

   Note: Ensure that the path specified with the -o option exists; otherwise **dbsvc** fails.

2. Start or stop the rshost service.

   From the Windows Services Control Panel:

- Locate the SQL Anywhere - SUPRelayServer service.
- To start or stop the service, right-click the service and choose the corresponding command.

From the command prompt:

- Change to `C:\inetpubs\wwwroot\ias_relay_server\Server`.
- To start the service, enter `dbsvc.exe -u SUPRelayServer`.
- To stop the service, enter `dbsvc.exe -x SUPRelayServer`.
- To uninstall the service, enter `dbsvc.exe -d SUPRelayServer`.
- To update the rshost with the latest relay server configuration, enter `rshost.exe –f rs.config –u`.

Tuning IIS for Improved Relay Server Performance

If relay server is installed to an IIS server, Sybase recommends tuning IIS to improve relay server performance. One best practice is to increase the number of connection pools used beyond just a single pool.

Tuning IIS allows you to connect more devices with greater success. Best practices for IIS tuning include:

1. Create separate application pools for `ias_relay_server\client` and `ias_relay_server\server`.

   Unwired Platform benefits from resource isolation between the 2 different connection pools. In IIS, associate `ias_relay_server\client` with `RS_CLIENT` and `ias_relay_server\server` with `RS_SERVER`.
2. Tune the `RS_CLIENT` by setting these properties:

   - Uncheck the **Request queue limit** box.
   - Increase **Web garden** size to approximately 5 x number of CPU core = 20.
3. Tune the `RS_SERVER` by increasing **Web garden** size, if the total number of RSOEs deployed is greater than 100. Otherwise, the default of 1 is sufficient.
4. If you anticipate a large number of HTTP responses, increase the `shared_mem` property to 128 MB in the `rs.config file` used by your relay servers. Alternatively, use the **mlmon** utility to increase the `buffer_size` property to a larger value.

Deploying Relay Server on IIS 6.x Web Servers

Use Microsoft IIS 6.x Web server to host relay server on any supported Windows OS host computers.

See also

- *Deploying Relay Server on IIS 7.x Web Servers* on page 55
- *Deploying Relay Server on Apache Web Servers* on page 62

Installing the Relay Server Components to IIS 6.x on Windows

After installing Sybase Unwired Platform, install the relay server component files to Microsoft IIS 6.*x* on Windows.

Prerequisites

Make sure Microsoft IIS 6.*x* is installed.

Task

1. Find the appropriate relay server ZIP file located on deployment edition of the installation media in `/modules/relayserver`, and copy it to your relay server system.

   There are two versions:
   - for 32-bit operating systems use `relayserver.zip` (but the name may vary depending on the platform used)
   - for 64-bit operating systems, use `relayserver_x64.zip`

   The ZIP file contains the directory `\ias_relay_server` with two subdirectories, `\Client` and `\Server`, which contain the required executables and DLLs for the relay server.
2. Extract the ZIP file under the Web site home directory that you use for the relay server, typically, `C:\Inetpub\wwwroot`.
3. Create an application pool:
   a) Start IIS Manager Console by opening a command prompt and running:
   ```
   mmc %systemroot%\system32\inetsrv\iis.msc
   ```
   b) Right-click **Application Pools** and create a new application pool, for example, RSOE_AP.
   c) Right-click the application pool and select **Properties** to edit the properties for the outbound enabler application pool. Select the **Performance** tab, and unselect **Shutdown Worker Processes After Being Idle**.
   d) In the **Recycling** tab, unselect **Recycle Worker Processes (In Minutes)**.
4. Enable Relay Service Web Extensions in the IIS Manager Console:
   a) Select the **Directory** tab to edit the properties of ias_relay_server. Set execute permissions to **Scripts And Executables**.
   b) Click **Create under Application Settings**.
   c) Select the application pool you created in step 3.
   d) Under **Web Service Extensions**, allow both `rs_server.dll` and `rs_client.dll` to be run as **ISAPI**.
5. Configure IIS for Unwired Platform device clients to communicate with relay server:
   a) Run IIS admin scripts, which are located in `\Inetpub\AdminScripts`.

b) Run the following console commands:

```
cscript adsutil.vbs set w3svc/1/uploadreadaheadsize 0
iisreset
```

If you do not perform this configuration step, you see this error message:

```
Could not connect to the Server. Session did not complete.
```

Configuring Relay Server to Run as a Windows Service

If you installed relay server on an IIS host, Sybase recommends that you set up the relay server as a Windows service to ensure that starts with the host.

1. At the command line, enter the following command, substituting all parameter values to match your configuration:

   ```
   dbsvc -as -s auto -w SUPRelayServer "C:\inetpub\wwwroot
   \ias_relay_server\server\rshost.exe" -q -qc -f c:\inetpub
   \wwwroot\ias_relay_server\server\rs.config -o c:\Sybase
   \logs\rs.log
   ```

   This command configures the relay server host process (rshost.exe) as a Windows service. By default, the **dbsvc** utility is installed in `<UnwiredPlatform_InstallDir>\Servers\SQLAnywhere11\BIN32\`.

   Note: Ensure that the path specified with the -o option exists; otherwise **dbsvc** fails.

2. Start or stop the rshost service.

   From the Windows Services Control Panel:

   - Locate the SQL Anywhere - SUPRelayServer service.
   - To start or stop the service, right-click the service and choose the corresponding command.

   From the command prompt:

   - Change to `C:\inetpubs\wwwroot\ias_relay_server\Server`.
   - To start the service, enter `dbsvc.exe -u SUPRelayServer`.
   - To stop the service, enter `dbsvc.exe -x SUPRelayServer`.
   - To uninstall the service, enter `dbsvc.exe -d SUPRelayServer`.
   - To update the rshost with the latest relay server configuration, enter `rshost.exe –f rs.config –u`.

Tuning IIS for Improved Relay Server Performance

If relay server is installed to an IIS server, Sybase recommends tuning IIS to improve relay server performance. One best practice is to increase the number of connection pools used beyond just a single pool.

Tuning IIS allows you to connect more devices with greater success. Best practices for IIS tuning include:

1. Create separate application pools for `ias_relay_server\client` and `ias_relay_server\server`.

   Unwired Platform benefits from resource isolation between the 2 different connection pools. In IIS, associate `ias_relay_server\client` with `RS_CLIENT` and `ias_relay_server\server` with `RS_SERVER`.
2. Tune the `RS_CLIENT` by setting these properties:

   - Uncheck the **Request queue limit** box.
   - Increase **Web garden** size to approximately 5 x number of CPU core = 20.
3. Tune the `RS_SERVER` by increasing **Web garden** size, if the total number of RSOEs deployed is greater than 100. Otherwise, the default of 1 is sufficient.
4. If you anticipate a large number of HTTP responses, increase the `shared_mem` property to 128 MB in the `rs.config file` used by your relay servers. Alternatively, use the **mlmon** utility to increase the `buffer_size` property to a larger value.

Deploying Relay Server on Apache Web Servers

Use Apache Web server to host relay server on Linux host computers.

1. *Installing Relay Server Components to Apache on Linux*

   After installing Sybase Unwired Platform, extract the relay server component files to the Apache modules directory, edit the main configuration file for the Apache HTTP server, and set the environment variables.
2. *Editing the Apache conf/httpd.conf File*

   Configure your relay servers to run in the Sybase Unwired Platform environment, by editing the Apache conf/httpd.conf file.
3. *Setting the Kernel Parameters*

   You need to modify the Linux kernel if the Relay Server controls more than the default number of 256 concurrent users. You must increase the ServerLimit and MaxClient values in the Apache conf/httpd.conf file before you set the kernel parameters. You can also modify other kernel values.

See also

- *Deploying Relay Server on IIS 7.x Web Servers* on page 55
- *Deploying Relay Server on IIS 6.x Web Servers* on page 59

Installing Relay Server Components to Apache on Linux

After installing Sybase Unwired Platform, extract the relay server component files to the Apache `modules` directory, edit the main configuration file for the Apache HTTP server, and set the environment variables.

1. Copy `relayserver_linux_x86.tar.gz` from `/modules/relayserver` on the installation media to the host machine. This file is available only on the installation media for the Enterprise Edition of Unwired Platform.
2. Extract the contents to the Apache `modules` directory.

   The file contains:
   - mod_rs_ap_client.so
   - mod_rs_ap_server.so
   - rs.config.sample
   - rshost
   - dblgen11.res
   - libdbtasks11_r.so
   - libdbicudt11.so
   - libdbicu11_r.so
   - libdblib11_r.so
   - dbsupport
3. Configure the relay server by generating a configuration file for it. Sybase recommends that you generate this file either using Sybase Control Center or the **regRelayServer** command line utility.

   By default, the file name is `rs.config` and you can output the file to place it in any convenient location.
4. Copy the generated file to the Apache `modules` directory.

   The relay server module expects `rs.config` to be in the same directory as `rshost`.
5. Set the PATH and LD_LIBRARY_PATH environment variables to include the Apache `modules` directory.
6. Edit the Apache `conf/httpd.conf` file.

   See *Editing the Apache conf/httpd.conf File*
7. When Apache spawns a process, determine if the following environment variables are set globally: $TMP, $TMPDIR or $TEMP.

   Make sure the Apache user process has write permissions to the `/tmp` directory.

   - If the environmental variables are set globally, you are finished.
   - If these variables are not set globally, or to save the default relay server log file in a specific temporary directory, edit the `/<apache-dir>/bin/envvars` file:

```
set TMP="/tmp"
export TMP
```

This command sets the environment variable in the shell that Apache creates before it spawns its processes.

Editing the Apache conf/httpd.conf File

Configure your relay servers to run in the Sybase Unwired Platform environment, by editing the Apache `conf/httpd.conf` file.

Key tasks for running the relay server with Sybase Unwired Platform include:

- Loading relay server client and server modules.
- Creating a <location> section for the client module.
- Creating a <location> section for the server module.
- Adding maximum server and client values on the Linux machine.

  Note: If you modify values in `conf/httpd.conf`, also change the kernel parameters (for Relay Server 11.0.x on Apache server only).
- Modifying the number of concurrent device users (or Web requests) supported in the production system.
- Setting global environment variables.

To edit the Apache `conf/httpd.conf` file:

1. Navigate to the Apache `modules` directory, and use a text editor to open the Apache `conf/httpd.conf` file.
2. Add these lines to load the relay server client and server modules:

   ```
   LoadModule iarelayserver_client_module modules/
   mod_rs_ap_client.so
   LoadModule iarelayserver_server_module modules/
   mod_rs_ap_server.so
   ```

   The client and server modules are invoked using different URLs. The client module explicitly looks for the string **iarelayserver** in the URL path.
3. Add these lines to create a <location> section for the client module:

   ```
   <LocationMatch /cli/iarelayserver/* >
        SetHandler iarelayserver-client-handler
   </LocationMatch>
   ```
4. Add these lines to create a <location> section for the server module, which corresponds the the URL suffix you configured for relay server:

   ```
   <Location /srv/iarelayserver/* >
        SetHandler iarelayserver-server-handler
        RSConfigFile "/<apache-install>/modules/rs.config"
   </Location>
   ```

   The RSConfigFile directive specifies the location of the relay server configuration file, `rs.config`. The `rs.config` file must reside in the same directory where the rshost

executable is deployed. For details about configuring this file, see *Configuring a Relay Server in a Multinode Cluster*.

5. For Relay Server 11.0.x on Apache server only, set the kernel on the Linux machine.

   Guidelines:
   - Set the `ServerLimit` and `MaxClient` values to greater than or equal to the number of UltraLite device clients. To calculate your number of clients: `(Number of UltraLite device clients) + (2 * Number of OE Backends)`.
   - The OE connections are considered clients from the Web server point of view. Each OE (backend) has an Up channel and a Down Channel (counting as two clients).
   - For example, if you have 500 concurrent UltraLite device clients and two backends, you would configure at least 504 MaxClients [`500 + (2 * 2)`]. For better performance and allow for margin, you might want to double that and configure `MaxClients` at 1000.

   a) Add the following lines to the file, and either use the default values, or change the values using the guidelines.

   ```
   ServerLimit 256
   MaxClient 256
   ```

   b) If you changed the `ServerLimit` and `MaxClient` default values, you will need to increase the semaphore sets. You may also want to configure additional kernel parameters on the Linux machine.

   Run **ipcs -l** to check the kernel limits value. You can use this information to modify kernel limits, as shown in the example below. For detailed information see *Setting the Kernel Parameters*.

   ```
   ------ Shared Memory Limits --------
   max number of segments = 4096
   max seg size (kbytes) = 4194303
   max total shared memory (kbytes) = 1073741824
   min seg size (bytes) = 1

   ------ Semaphore Limits --------
   max number of arrays = 640
   max semaphores per array = 250
   max semaphores system wide = 160000
   max ops per semop call = 32
   semaphore max value = 32767

   ------ Messages: Limits --------
   max queues system wide = 16
   max size of message (bytes) = 65536
   default max size of queue (bytes) = 65536
   ```

6. By default , Relay Server on Linux is configured for 100 concurrent device users (or Web requests) in the production system. If you have more than this, modify the number to an appropriate value in this section:

   ```
   #
   # MaxKeepAliveRequests: The maximum number of requests to allow
   ```

```
# during a persistent connection. Set to 0 to allow an unlimited
amount.
# We recommend you leave this number high, for maximum
performance.
#
MaxKeepAliveRequests 100
```

Setting the Kernel Parameters

You need to modify the Linux kernel if the Relay Server controls more than the default number of 256 concurrent users. You must increase the `ServerLimit` and `MaxClient` values in the Apache `conf/httpd.conf` file before you set the kernel parameters. You can also modify other kernel values.

Modify kernel:

1. To display the current values of the semaphore kernel parameters, use:

   ```
   cat /proc/sys/kernel/sem
   ```

   ```
   250 32000 32 4096
   ```

   The four values displayed are:
   - `SEMMSL` – maximum number of semaphores per set or array.
   - `SEMMNS` – maximum number of semaphores system–wide. This values is determined by `Maximum Number of Arrays * Maximum Semaphores/array`.
   - `SEMOPM` – maximum number of operations allowed for one semop call.
   - `SEMMNI` – maximum number of semaphore identifiers (sets).
2. Open the `/etc/sysctl.conf` file to edit.
3. Update the `/etc/sysctl.conf` file:
   a) You can modify the semaphore value, based on your system configuration, for example: `kernel.sem = 250 160000 32 640`.

      Examples for increasing the number of "semaphore sets" (`semmni`) in the system:
      - Example 1 – Increasing semaphore sets to 10 times the current number of existing sets in the system. Note that the second number in the group should ALWAYS be equal to the multiplication of the first number by the last number `(max semaphores per array) * ((max number of arrays) * 10 times)`, for example, `250*1280 = 320000`. You would enter this semaphore value in the `sysctl.conf` file:

        ```
        kernel.sem = 250 320000 32 1280
        ```
      - Example 2 – Assume the Apache Web server is configured with 500 worker processes and the Relay Server has 3 backend servers configured. The total number of semaphore sets needed is `500 + (4*3) = 512`. Assume that 128 was the default number of semaphore sets in the system, then the new total number of semaphore sets is `128 + 512 = 640`. Assume that 250 was the default number of semaphores per array, then new total number of semaphores system wide is `250`

`* 640 = 160000`. You would enter this semaphore value in the `sysctl.conf` file:

```
kernel.sem = 250 160000 32 640
```

b) You may also want to configure additional kernel values on the Linux machine. Following are sample maximum limits:

```
------ Shared Memory Limits --------
max number of segments = 4096
max seg size (kbytes) = 4194303
max total shared memory (kbytes) = 1073741824
min seg size (bytes) = 1

------ Semaphore Limits --------
max number of arrays = 640
max semaphores per array = 250
max semaphores system wide = 160000
max ops per semop call = 32
semaphore max value = 32767

------ Messages: Limits --------
max queues system wide = 16
max size of message (bytes) = 65536
default max size of queue (bytes) = 65536
```

4. Save the file and reboot the machine.
5. Run `sysctl` with `-p` parameter to load in `sysctl` settings from the default file `/etc/sysctl.conf`.

   ```
   sysctl -p
   ```

   The entries from the `sysctl.conf` file are read during startup by the network initialization script. On some distributions you may be required to add `sysctl -p` in one of the system initialization files (for example, `rc.local`) so that kernel parameters are set after each reboot.

Configuring a Multinode Relay Server Cluster with Sybase Control Center

A multinode relay server cluster includes one or more relay server nodes and multiple RSOEs deployed to corresponding Unwired Server nodes. Use Sybase Control Center to make a multinode relay server cluster operational

Alternatively, if you have an extremely large relay server environment, you can alternately use the `<UnwiredPlatform_Installdir>\UnwiredServer\config\rsoe.config.template.xml` and the `regrelayserver.bat` script to deploy these components.

See also

- *Relay Server (rs.config) Configuration File* on page 364
- *Installing Relay Server to Multiple Web Server Host Nodes* on page 55
- *Enabling Application Request Routing for IIS Relay Server Hosts* on page 69

Setting and Distributing Properties for Multiple Relay Servers

Initially configure the first relay server node in Sybase Control Center to simplify setting relay server property values and generating a configuration file from those values. Administrators can then easily transferred configurations to other relay server hosts on the DMZ.

Prerequisites

Ensure that the cluster and Unwired Servers are registered in Sybase Control Center. If the cluster name or the server name does not appear in the navigation pane, you must register these artifacts with Sybase Control Center. You must also start Unwired Servers in the cluster you are configuring.

Task

1. In the Sybase Control Center navigation pane, click the cluster name, then click the **Relay Server** tab.
2. Register and configure the first relay server:
   a) Click **New**.
   b) Configure all pages of the wizard, then click **Finish**. The relay server is registered and these properties are stored in the cluster database. See *Creating a Custom Relay Server Configuration* in the Sybase Control Center online help.
3. Configure remaining relay server nodes:
   a) Export the properties configured for the first node by selecting the first relay server node entry that appears in the Relay Server tab, then clicking **Generate**.
   b) When prompted, choose **Relay server configuration properties file**, and generate the entire file, so that new relay server nodes can be configured with it.
   c) Manually edit this file, to set the unique connection properties for the new server, as well as other enhancements you need to make that are not supported by Sybase Control Center.

      For complete reference details on all configuration properties available, see *http://infocenter.sybase.com/help/index.jsp?topic=/com.sybase.help.sqlanywhere.11.0.1/mlserver_en11/ml-relayserver-config-file.html*.
   d) Copy the edited file to the appropriate Web server location.

      The path depends on the host type:
      - For IIS: `C:\Inetpub\wwwroot\IAS_relay_server\Server\rs.config`
      - For Apache: `<Apache_Home>/modules/rs.config`
4. To upload these configuration changes to relay server:

```
rshost –u –f <WebServerPath>rs.config
```

Next

Upon completion, administrators still need to create one or more RSOEs for each server node using Sybase Control Center.

See also

- *Register Relay Server (regRelayServer) Utility* on page 330

Setting Up RSOEs for Server Farm Nodes

Configure three RSOEs for each server node you added to each relay server farm you defined in the relay server configuration.

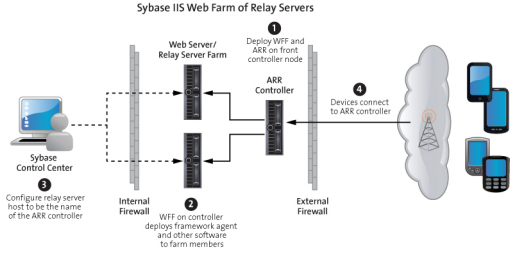
1. In the navigation pane of Sybase Control Center, click **Servers** > <**ServerNode**> > **Server Configuration**.
2. In the administration pane, select the **RSOE** tab, then click **New**.
3. Conifigure the RSOE:
   - General properties – to define the runtime context in which the RSOE process operates.
   - Start options – to set options that are enabled automatically each time an RSOE process starts.

   See *Setting Up RSOE* in Sybase Control Center online help.
4. Start the RSOE, by clicking the name of the newly created RSOE in the RSOE tab and click **Start**.
5. Repeat steps 1-4 until all required RSOEs are configured and started on each server node in your environment.

Enabling Application Request Routing for IIS Relay Server Hosts

Sybase recommends using IIS Application Request Routing (ARR) to balance request loads among IIS instances that host relay servers.

When you complete this task, your completed topology appears similar to this:



Like this image suggests, the first step in the task is to install and set up the IIS ARR controller. Continue all steps until your device client can connect to the environment using the ARR controller connection properties.

1. *Installing Required Microsoft Components*

Prepare a new standalone host by installing required components that make up the load balancer on supported Windows operating systems. Install only named components and install them in the documented order. Do not share this host with other Unwired Platform components.

2. *Configuring the Relay Server IIS Host for Load Balancing*

   Update the configuration of the IIS Web server used by relay server already deployed to the DMZ so that load balancing operates correctly.

3. *Connecting Relay Server to the Application Request Routing Server*

   The relay server needs to be able to receive in-bound client device requests from the ARR service. To enable that, you must identify the ARR host in the relay server configuration.

4. *Connecting Device Clients to the Application Request Routing Server*

   With the ARR load balancer and relay servers set up and available, pass connection details to the developers responsible for configuring client device connection parameters.

See also

- *Load Balancing* on page 10
- *Configuring a Multinode Relay Server Cluster with Sybase Control Center* on page 67
- *Deploying Relay Server on IIS 7.x Web Servers* on page 55

Installing Required Microsoft Components

Prepare a new standalone host by installing required components that make up the load balancer on supported Windows operating systems. Install only named components and install them in the documented order. Do not share this host with other Unwired Platform components.

You must use the Microsoft Web Platform Installer (Web PI) 3.0 only. Web PI is a free tool.

1. Download and install the Microsoft .NET Framework.
2. Download Microsoft Web PI 3.0.

   You can download the installer from *http://www.microsoft.com/web/downloads/platform.aspx*.

3. Install Web Platform components in this order:
   a) Web Farm Framework 2.0.

      This component allows you to create Web server farms for all IIS servers on your network.

   b) Application Request Routing 2.1 with HotFix for External Cache 1.0.

      This component allows you to route requests and balance loads among IIS servers identified in the farm you set up.

4. Click **Exit** to close the Web Platform Installer.

Configuring the Relay Server IIS Host for Load Balancing

Update the configuration of the IIS Web server used by relay server already deployed to the DMZ so that load balancing operates correctly.

You must perform these updates with the IIS Management Console.

See also

- *Connecting Relay Server to the Application Request Routing Server* on page 73

Configuring IIS for the Relay Server Load Balanced Environment

Configure the IIS server that hosts relay server to work with Application Request Routing (ARR).

Disable these two IIS features to support ARR:

- Idle time-outs – Because all HTTP requests go through ARR, the worker process for the Default Web Site must always be running. You can configure the worker process to always run by disabling the Idle Time-Out setting in the application pool process model for the relay server.
- Application pool recycling – Prevent an application pool from recycling unexpectedly.

1. Open IIS management console.
2. In the Connections pane, select **Application Pools** then choose **DefaultAppPool**.
3. Click **Actions**, then **Advanced Settings**.
4. Change the **Idle Time-out (minutes)** value to `0`.
5. Click **Actions**, then **Recycling**.
6. Uncheck **Regular time intervals (in minutes)**.

Setting Up the IIS Web Server Farm

Set up an IIS Web server farm to identify all IIS servers used to host relay servers on your network. Once the farm has been defined, you can balance loads among servers in this farm.

Add all IIS servers that you intend to manage as a farm.

1. In the **Connections** pane of the IIS Management console, right-click **Server Farm**, then choose **Create Server Farm**.
2. Configure the farm properties by:
    - Setting a name. Choose a name that identifies the farm as a relay server farm for Unwired Platform.
    - Checking the **Server farm is available for load balancing** and **Provision server farm** boxes.

- Configuring an administrator account. Use the platform administration credentials to make all relay server hosts availabe from the IIS Management console.

3. Add IIS servers by setting the server names and identifying which server is a primary server, then click **Finish**.

   The management console connects to the servers and begins to automatically provision the IIS servers with the Web Farm Framework. No manual installation is required.
4. Validate the completion of the provisioning:
   a) In the **Connections** pane, click **Server Farms** > ***MyFarmName*** > **Servers**.
   b) Check the trace messages to see when the operation has completed.

Setting Up Application Request Routing Rules and Load Balance Policies

Set up Application Request Routing (ARR) rules to provide load balancing for IIS servers used as relay server hosts. Use load balance policies to configure which algorithm the ARR server should use.

1. On any IIS server that hosts a relay server, configure routing rules:
   a) Use the IIS Management console to open Routing Rules for the Web Server Farm.
   b) Check these boxes:
      - **User URL Rewrite** to inspect incoming requests and rewrite application request URLs as required.
      - **Enable SSL offloading** to perform all communication between the ARR server and the relay servers in clear text, even for HTTPS requests from clients to the ARR server. When both the ARR server and the relay servers are deployed within a trusted network, such as within the same datacenter, enabling SSL offloading does not sacrifice security. Also, enabling this feature can further help to maximize the server resources on the application servers, since they do not have to spend cycles in encrypting and decrypting requests and responses.
2. Set up the load-balancing policy:
   a) Use the IIS Management console to open Load Balancing for the Web Server Farm.
   b) Change the default algorithm to `Weighted round robin`.
3. Validate that the Web server farm is set up for load balancing:
   a) In the IIS Management console **Connections** pane, click **Server Farms** > ***MyFarmName*** > **Servers**.
   b) In the Servers list, check that all servers display `Yes` in the **Ready for Load Balancing** column.
4. Verify that requests are balanced equally among the relay servers:
   a) In the IIS Management console's **Connections** pane, click ***MyFarmName***.
   b) Double-click Monitoring and Management.
   c) Review the **Load Balancing** column's percentages and ensure each server is equally weighted.

Connecting Relay Server to the Application Request Routing Server

The relay server needs to be able to receive in-bound client device requests from the ARR service. To enable that, you must identify the ARR host in the relay server configuration.

1. In the Sybase Control Center navigation pane, click the cluster name, then click the **Relay Server** tab.
2. Update the configuration of relay servers:
   a) Click the name of the relay server.
   b) Click **Properties**.
   c) For the **Host** property, change the value from the relay server name to the IP address or DNS name for the of the ARR server acting as the load balancer. Client devices must be able to use the value entered here.

See also

- *Configuring the Relay Server IIS Host for Load Balancing* on page 71

Connecting Device Clients to the Application Request Routing Server

With the ARR load balancer and relay servers set up and available, pass connection details to the developers responsible for configuring client device connection parameters.
Ensure all connections properties use the ARR server host, rather than the relay server host typically used.

Scaling an Existing Relay Server Cluster

If your environment needs to grow or reduce in response to changes in user demand or business requirements, scale your architecture in or out horizontally, by adding or removing relay server nodes and RSOE processes as required.

Sybase recommends that you add or remove relay servers or RSOEs with Sybase Control Center.

If you have an extremely large relay server environment, you can use the `<UnwiredPlatform_Installdir>\UnwiredServer\config\rsoe.config.template.xml` and the `regrelayserver.bat` script to add relay servers to the cluster. However, removal capabilities with this script are limited to:

- Removing RSOE configurations, but not relay server configurations.
- Removing all local RSOEs, but not some of them

Adding or Removing a Relay Server with Sybase Control Center

Scale an existing relay server cluster by adding more relay server nodes to the existing production architecture.

1. In the Sybase Control Center navigation pane, click the cluster name for which relay servers are required, then click the **Relay Server** tab.
2. If you need to add a new relay server:
   a) Click **New**.
   b) Configure all pages of the wizard then click **Finish**. The relay server is registered and these properties are stored in the cluster database. See *Creating a Custom Relay Server Configuration* in the Sybase Control Center online help.
3. If you need to remove a relay server, select the relay server name and click **Delete**.

   Deleting a relay server only removes it from Sybase Control Center and unregisters it from the cluster database.
4. To update existing configuration files to hold new relay server node definitions:
   a) Export the properties by selecting the new relay server node entry in the Relay Server tab, then clicking **Generate**.
   b) When prompted, generate the whole file.

      The farm information is updated when you apply the changes in the last step.
   c) On each relay server node, copy and paste these lines from the generated file into the configuration file.

      The path depends on the host type:
      - For IIS: `C:\Inetpub\wwwroot\IAS_relay_server\Server\rs.config`
      - For Apache: `<Apache_Home>/modules/rs.config`
5. To remove properties of deleted relay servers, manually delete the definition from the configuration file.
6. Save all edits.
7. To upload these configuration changes to the relay server run:

   ```
   rshost –u –f <WebServerPath>rs.config
   ```

Adding or Removing an RSOE with Sybase Control Center

Scale an existing relay server cluster by adding or removing RSOEs as required. Sybase recommends at least three RSOEs per Unwired Server node for each synchronization type (MBS and RBS).

Add more RSOEs when you :

- Deploy a new Unwired Server
- Are supporting a new synchronization service (MBS/RBS)

- Have a bottleneck (high CPU usage) against an RSOE process for which you need to improve throughput

Remove RSOEs when:

- The recommendation proves to be more than your user demand requires
- A deployed Unwired Server is retired from the production environment

1. In the navigation pane of Sybase Control Center, click **Servers** > ***ServerNode*** > **Server Configuration**.
2. In the administration pane, select the **RSOE** tab.
   - To add an RSOE to a cluster, then click **New**, then configure its properties (General and Start Options). See *Setting Up RSOE* in the Sybase Control Center online help.
   - To remove an RSOE from a cluster, click the RSOE you want to remove, stop the process, then click **Delete**.

Upgrading an Existing Relay Server Cluster

Manually upgrade relay server clusters to the current version if you did not use the installer.

Updating Relay Server Binaries for IIS on Windows

Update previously installed relay server binaries on Microsoft Internet Information Service (IIS), so they match the version used by the current version of Unwired Platform. Sybase recommends that you always keep binaries current.

1. From the installation media, save the relevant relay server ZIP file to the relay server host:
   - For 32-bit operating systems, use `relayserver.zip`
   - For 64-bit operating systems, use `relayserver_x64.zip`

   The ZIP file contains the required executables and DLLs for the relay server.

   Note: These files are only available on the installation media for the deployment edition of Unwired Platform.
2. Stop IIS and relay server, for example:

   ```
   C:\inetpub\wwwroot\ias_relay_server\Server > dbsvc -x
   SUPRelayServer
   ```
3. Extract the ZIP file under `C:\Inetpub\wwwroot\` to create the folder structure:

   ```
   C:\inetpub\wwwroot\ias_relay_server
   C:\inetpub\wwwroot\ias_relay_server\Server
   C:\inetpub\wwwroot\ias_relay_server\Client
   ```
4. Verify that all the `DLL` files copied under these folders are of the required version, for example, 11.0.1.2527.

5. Restart the relay server service, for example:

   ```
   C:\inetpub\wwwroot\ias_relay_server\Server > dbsvc -u
   SUPRelayServer
   ```

6. Check the relay server logs to make sure the first line reflects the new version. For information on where log files are output, see *Log File Locations* in the *Troubleshooting* guide.

   ```
   I. 2010-12-04 07:22:28. < 3160.- > iAnywhere Relay Server Version
   11.0.1.2527
   I. 2010-12-04 07:22:28. < 3160.- >
   I. 2010-12-04 07:22:28. < 3160.- > Copyright © 2001-2009,
   iAnywhere Solutions, Inc.
   I. 2010-12-04 07:22:28. < 3160.- > Portions copyright © 1988-2009,
   Sybase, Inc. All rights reserved.
   I. 2010-12-04 07:22:28. < 3160.- > Use of this software is
   governed by the Sybase License Agreement. Refer to http://
   www.sybase.com/softwarelicenses
   ...
   ```

See also

- *Updating Relay Server for Apache on Linux* on page 76
- *Removing RSOE and Migrating Configuration Files With Scripts* on page 77

Updating Relay Server for Apache on Linux

Update previously installed relay server binaries on Apache, so they match the version used by the current version of Unwired Platform. Sybase recommends that you always keep binaries current.

1. Copy `relayserver_linux_x86.tar.gz` from `/modules/relayserver` on the installation media to the host machine.

   Note: This file is available only on the installation media for the deployment edition of Unwired Platform.

2. Stop Apache and relay server.
3. Extract the contents of the archive to the Apache `modules` directory.
4. Verify that all the modules and library files copied under these folders are of the required version, for example, 11.0.1.2527.
5. Restart Apache and relay server.
6. Check the relay server logs to make sure the first line reflects the new version. For example, from `C:\windows\System32\inetsrv\ias_relay_server_host.log`:

   ```
   I. 2010-12-04 07:22:28. < 3160.- > iAnywhere Relay Server Version
   11.0.1.2527
   I. 2010-12-04 07:22:28. < 3160.- >
   I. 2010-12-04 07:22:28. < 3160.- > Copyright © 2001-2009,
   iAnywhere Solutions, Inc.
   ```

```
I. 2010-12-04 07:22:28. < 3160.- > Portions copyright © 1988-2009,
Sybase, Inc. All rights reserved.
I. 2010-12-04 07:22:28. < 3160.- > Use of this software is
governed by the Sybase License Agreement. Refer to http://
www.sybase.com/softwarelicenses
...
```

See also

- *Updating Relay Server Binaries for IIS on Windows* on page 75
- *Removing RSOE and Migrating Configuration Files With Scripts* on page 77

Removing RSOE and Migrating Configuration Files With Scripts

When upgrading to this version of Unwired Platform, you must remove existing RSOE services and migrate configuration files to XML. If you did not use the installer to perform this upgrade, you must manually upgrade the RSOE and configuration files.

Prerequisites

Ensure the `relayserver.properties` file exists and that it contains the correct relay server/relay server outbound enabler configuration information used to configure Unwired Server 1.5.5 relay server outbound enablers. Do not modify the relay server environment while this script is running (that is, do not add/remove a relay server or outbound enabler).

Task

Upgrading RSOE requires that you manually remove the RSOE service and replace it with a server process that you must then start. Migrating configuration files safely extracts values from the `relayserver.properties` file and inserts them into cluster database. Sybase recommends that you perform upgrade and migration with the product installer. However, we provide these steps in case some manual action is needed.

1. Remove the existing RSOE service by running this command:

   ```
   sc delete <ServiceName>
   ```

2. Ensure the cluster and consolidated databases are running.
3. On each server node with one or more RSOE, run this command:

   <UnwiredPlatform_InstallDir>\Servers\UnwiredServer\bin\regrelayserver.bat migrate

   ```
   <UnwiredPlatform_InstallDir>\Servers\UnwiredServer\bin
   \regrelayserver.bat -migrate
   ```

4. Repeat these steps on each server node to upgrade all RSOEs and migrate all configuration files.
5. Upon a successful upgrade, restart RSOEs in Sybase Control Center or restart the host computer on which the RSOEs have been installed.

   For details about starting RSOEs from Sybase Control Center, see *Starting and Stopping RSOE* in the Sybase Control Center online help.

See also

- *Updating Relay Server Binaries for IIS on Windows* on page 75
- *Updating Relay Server for Apache on Linux* on page 76

Subscribing and Connecting to Sybase Hosted Relay Service

Subscribe online to the Sybase Hosted Relay Service, and configure your environment with information provided during subscription, rather than installing a relay server on your network.

A hosted relay server is typically used by pilot, development, or test environments. All other environments should create a new relay server cluster in Sybase Control Center and use relay server binaries on their DMZ.

1. Register Unwired Server with the relay service.
   a) Register or log in to the Sybase Hosted Relay Service at *https://relayserver03.sybase.com/ias_relay_server/account/index.php*.

      Complete any mandatory fields, then click **Submit**.
   b) From the navigation bar on the left, click **Manage Account**.
   c) Click **Add New Sybase Unwired Platform Farm**.

      Create one or more farms as required by your development or test environment.
      - Select at least one farm type.
      - Do not select **DCN farm**.
      - Select **MBS farm** for Hybrid Web Container applications.
      - Enter the farm name, which serves as the farm ID in Sybase Control Center. The suffix `RBS` or `MBS` is appended to the end of the farm name.
      - Enter the server name, which is used as the server node ID in Sybase Control Center. The server name must contain only alphanumeric characters.
   d) Click **Create Farm**.
   e) Click **Configuration Instructions** from the confirmation message.

      Keep this page open, or make a copy of these details so you have them available for further configuration tasks in Sybase Control Center and on the client devices.
2. Launch Sybase Control Center to register the relay service and create Relay Server Outbound Enablers (RSOEs) for it.
   a) Click the Unwired Platform cluster for which you are configuring the relay server, select the **Relay Servers** tab, then click **+ New**.
   b) Enter the general configuration information from the configuration instructions, then click **Next**.

Host	relayserver.sybase.com
HTTP port	80
HTTPS port	443

c) Enter the server farm information.

Farm ID	Copy the farm name from configuration instructions
Type	Messaging (MBS) or Replication (RBS)

d) Click +, then click the farm ID field.
e) As the **Node ID**, use the server name you registered in the Sybase Hosted Relay Service.
f) Copy the **Token** from the configuration instructions and paste it.
g) Click **+** to add the server node to the list, then **Finish**.

3. Add the relay server as an RSOE.
   a) In the left pane, under **Servers** > <***Unwired Server name***>, select **Server Configuration**.
   b) Select the **RSOE** tab.
   c) Click **+ New** to add the relay server as an RSOE.
   d) Select the details from the drop-down lists, then click **Finish**.
4. Start all RSOEs.

   Select some or all RSOEs, then click **Start**. Check the Status column to verify if the status changes to `Running`.
5. Select one or more RSOEs, then click **Retrieve Log** and review messages to ensure the RSOE is running correctly.
6. Once the environment is functional, keep the connection properties that are displayed on the Configuration Instructions page to share with developers and device users.

   Developers need these values to configure cluster information in an Unwired Server connection profile, and to set values in the Connection screen of a device application. Values must match or the connection to the relay server enabled runtime fails.

CHAPTER 6 Security Administration

Because Unwired Server coordinates data between enterprise information system data sources and device clients, it is likely that information is proprietary and confidential. Therefore, the data needs protection.

Unwired Platform has several features to assist in building a secure end-to-end environment. Understand what features exist before you begin configuring and implementing various security mechanisms.

Security Layers

The platform administrator is responsible for most aspects of security, however application security can be managed by domain administrator.

Security in Unwired Platform can be separated into distinct areas:

- Transport security – secures client/server communications by using digital certificates and public-key cryptography. Public-key cryptography is a widely used approach that secures communication between source and destination components as data is transferred across public or private networks.
- User security – authenticates and authorizes user or administrator access by using the services of a configured provider listed in a security configuration. Once authenticated, access is given to perform operations with a package. However, application data access then further depends on the logical role assigned to the MBO and operations, and its mapping to a role/group in the provider.

  Unwired Server relies on user entries that are stored in a existing directory like LDAP. When a user requests data in the device application, the application uses Unwired Server to authenticate the user and authorize access to specific resources with configured providers.
- Network security – secures components behind a firewall using a proxy. Unwired Platform components are usually installed in a private network, thereby isolating it from external network by one or more intermediary networks. Typically, this involves separating Unwired Servers from clients, either because of firewall policies or address space concerns. You then use a proxy server outside of the firewall (for example, the relay server) to open ports that allow secure data passage through the firewalls to Unwired Platform components.

  Most Unwired Platform installations will use the Relay Server as the network proxy in production scenarios. The Relay Server proxy tunnels between network zones; you can chain multiple servers together to allow for multiple hops. This network architecture allows the Unwired Server to connect and communicate with remote client devices across

multiple network boundaries (for example, the Internet, wireless network, or even other private networks).

- Data security – ensures that data is kept safe, and that access to it is controlled to protect data and keep it private. Data security has two sides:
  1. Data security – can be secured in platform databases by changing the DBA password and enabling encryption. Client-side databases can be encrypted using the Unwired Platform API.
  2. Device security – the ability to lock or wipe the device of enterprise data if the device is lost or stolen, with the purchase of an Afaria license.

Together, these levels provide multitiered and complete protection in an Unwired Platform environment — especially when used with a relay server. Relay server (as opposed to a typical proxy server) allows secure data communication across network zones without opening holes in firewalls. This makes relay server more secure than the average proxy server counterparts.

Transport Security Setup

Device client communications, back-end server communications, and Unwired Platform system server communications can all be encrypted.

To fully secure the network communications of all components in a Unwired Platform enabled production environment, you need to enable encryption in the following areas of an Unwired Platform-enabled production environment:

- Front-line data exchange: Encrypting replication based synchronization data that is transfered between device clients and Unwired Platform components. For RBS device clients, you also have to install a keystore with the HTTPS server's keys on the client as well. For MBS device clients, data is always encrypted and does not require any further configuration.
- Back-end server data exchange: Sending data change notifications by using a secure interface that notifies Unwired Server of changes to EIS data.
- Administrative data: Encrypting data exchanged among Sybase Control Center and Unwired Server.

Protocol and Component Reference

Review this summary of encrypted protocols used in Unwired Platform, and how each component is configured to enable encryption.

Transport stream	Encrypted protocols supported	Configuration
Data change notifications (DCNs) between an enterprise information server (EIS) and Unwired Server	HTTPS using SSL	(EIS developer) use HTTPS to construct and send DCN requests to the listener. (Administrator) generate certificate, importing certificate into server-side keystore, then using Sybase Control Center to configure an HTTPS listener for DCNs in the Unwired Server configuration properties.
Unwired Server and Sybase Control Center	IIOPS using SSL	Generate the certificate, import the certificate into the Unwired Server key store, import the public certificate into the Sybase Control Center key store, configure IIOPS in Sybase Control Center, in the Unwired Server configuration properties, then update the Sybase Control Center server resource properties to use the IIOPS port and secure mode.
Unwired Server and relay server for RBS synchronization (MBS data is exchanged using a proprietary encryption over HTTP, so there is no need to use SSL.)	HTTPS with mutual authentication is available only for RBS on Windows Mobile. HTTPS without mutual authentication is available for all RBS clients.	On the relay server host: configure the Web server used for the relay server (IIS or Apache) to use a secure port, then configure relay server protocols, ports and certificates in Sybase Control Center. On the client (in the case of mutual authentication): Install a `trusted-Certificates` file that contains a root certificate for the relay server's certificate.

Note: In addition to configuring each component to use the correct protocol and port, you must secure all of your generated server/client certificates.

Security Profiles and Listeners

In Sybase Control Center, HTTPS and IIOPS listeners are configured then assigned to the security profiles by a platform administrator. There are two kinds of listeners that can be configured and assigned: HTTPS listeners (for DCNs) and IIOPS listeners for server administration.

Security profiles define the security characteristics of a client/server session. Assign a security profile to a listener, which is configured as a port that accepts client connection requests of various protocols. Unwired Server uses multiple listeners. Clients that support the same characteristics can communicate to Unwired Server via the same port defined in the listener.

Before you create a security profile in Sybase Control Center as part of the Unwired Server configuration, you must always create key pair of public and private keys and install that key pair in the keystore used by Unwired Server. You can then complete the security profile setup by specifying the key file, alias (also called a label) and keystore password.

Once you have created the security profile, configure the listener to use a security profile communicating over the encrypted protocol.

Configuring Security Profiles

Configure security profiles to secure communication between Unwired Server administration and DCNs.

Prerequisites

Before creating a security profile, ensure that you possess digital certificates that have been verified and signed by third-party trusted authorities, as well as import required certificates in to the Unwired Server keystore. See the next topic, *Security Key and Certificate Basics*.

Task

1. In the left navigation pane, expand the **Servers** folder and select a server.
2. Select **Server Configuration**.
3. In the right administration pane, select the **General** tab.
4. From the menu bar, select **SSL Configuration**.
5. Create a new Security Profile:
   a) Name the security profile.
   b) Enter the case sensitive certificate alias for the profile (defined in the server keystore).
   c) Select the Authentication option.
6. Click **Save**.
7. In the server restart dialog, click **OK**.
8. Restart the server for these changes to take effect.

Next
Use the profile to encrypt administration and DCN ports.

Security Key and Certificate Basics

You can configure Unwired Server to accept connections over the secure protocols IIOPS and HTTPS by managing certificates and keys in the Unwired Platform security repository.

.

Table 7. Key and Certificate Management in Unwired Platform

Item	Details
Private key and certificate types	Supported types are “pkcs12” and “jks.”
Utility to maintain keystores and truststores	The SUN Java keytool utility, or the Sybase **createkey** utility.
Certificate assignment to a listener	In Sybase Control Center using a security profile you create. Ensure that the certificate has been imported into the Unwired Server keystore.
Location of the keystore and truststore	In `<UnwiredPlatform_Install-Dir>\Servers\UnwiredServer\Repository\Security`, use `keystore.jks` for the keystore and `truststore.jks` for the truststore. The keystore holds all server-side certificates (private keys); the truststore holds the trusted certificate authority (CA) root certificates.

Encrypting Unwired Server Administration Connections

Enable Unwired Server administration encryption to protect administration metadata that is transferred between Unwired Servers and Sybase Control Center.

Setting up administration encryption requires that you create certificates and import them into their respective keystores, and then use IIOPS to connect to a specified port. Sybase recommends that you create a security profile to associate a certificate with this port. You can get a certificate from a certificate authority (CA), or you can use the **createcert** utility to generate a self-signed certificate.

Configuring Unwired Server Administration Certificates

By default, Unwired Server includes two security profiles: "default" and "default_mutual." The "default" security profile uses the "sample1" certificate and the "default_mutual" security

profile uses the "sample2" certificate. The certificate used by any other user-created security profiles is specified during security profile creation.

Prerequisites

- Determine whether the SSL security profile for the secure management (IIOPS) port requires "default" or "default_mutual" authentication. If the security profile requires "default" authentication, the Sybase Control Center truststore should contain the Unwired Server certificate. If it requires "default_mutual" authentication, the Sybase Control Center and Unwired Server truststores should each contain a copy of the other's certificate.
- By default, the keystore and truststore passwords are both "changeit". In a production deployment of Unwired Server, use the **keytool** utility to set new passwords for both of these files.

Task

These steps describe the basics of exporting and import a certificate. Use the same steps to import your certificate into Unwired Server and Sybase Control Center keystore.

1. Set up the certificates:
   a) Add %JAVA_HOME% to your system path.
   b) At a command prompt, change to `<UnwiredPlatform_InstallDir>\Servers\UnwiredServer\Repository\Security`.
   c) Export the "default" security profile certificate `sample1.crt` from the Unwired Server keystore by running:

      ```
      keytool -keystore keystore.jks -storepass changeit -alias
      sample1 -exportcert -file sample1.crt
      ```

   You can also create and import new certificates. For more information about how to generate a new keystore and truststore, see *http://docs.sun.com/app/docs/doc/819-3658/ablrb?a=view*.
2. Configure Sybase Control Center:
   a) Open `<UnwiredPlatform_InstallDir>\SCC-XX\services\Messaging\lib\eas\lib\Repository\Server\EmbeddedJMS\Instance\com\sybase\djc\server\ApplicationServer\EmbeddedJMS.properties`.
   b) Insert these values to the keystore and truststore files :

      ```
      keyStore=<filePath>/<keyStoreName>.jks
      keyStorePassword=<password>
      trustStore=<filePath>/<trustStoreName>.jks
      trustStorePassword=<password>
      ```

      Or, you can copy the Unwired Server keystore and truststore files and use them for Sybase Control Center instead.

c) Import `sample1.crt` into the Sybase Control Center keystore by running:

```
keytool -keystore keystore.jks -storepass changeit -alias
sample1 -importcert -file sample1.crt
```

3. If you are running multiple Unwired Servers as part of a clustered environment, ensure these changes are repeated on all nodes. If certificates are issued to a specific host, you must generate a new certificate for each node.

Next

Enable and configure the administration port to use the security profile in Sybase Control Center.

See also

- *Key Creation (createkey) Utility* on page 337
- *Encrypting DCN Connections* on page 98
- *Data Change Notifications* on page 195

Enabling and Configuring Administration Encryption for Unwired Server

Enable encryption to securely transfer data between the Unwired Server administration listener and Sybase Control Center.

You can create or change a security profile that saves SSL setup data for a particular server instance. Using the security profile, you associate a specific key with the encrypted port.

1. In the left navigation pane, expand the **Servers** folder and select a server.
2. Select **Server Configuration**.
3. In the right administration pane, click **General**.
4. Optional. If you want to create a new security profile, select **SSL Configuration**.
5. In the **Configure security profile table**:
   a) Enter a name for the security profile.
   b) Enter a certificate alias. This is the logical name for the certificate stored in the keystore.
   c) Select an authentication level:

      If the security profile authenticates only the server, then only the server must provide a certificate to be accepted or rejected by the client. If the security profile authenticates both the client and the server, then the client is also required to authenticate using a certificate; both the client and server will provide a digital certificate to be accepted or rejected by the other.

Profile	Authenticates	Cipher suites
intl	server	• SA_EX-PORT_WITH_RC4_40_MD5 • RSA_EX-PORT_WITH_DES40_CBC_SHA
intl_mutual	client/server	• RSA_EX-PORT_WITH_RC4_40_MD5 • RSA_EX-PORT_WITH_DES40_CBC_SHA
strong	server	• RSA_WITH_3DES_EDE_CBC_SHA • RSA_WITH_RC4_128_MD5 • RSA_WITH_RC4_128_SHA
strong_mutual	client/server	• RSA_WITH_3DES_EDE_CBC_SHA • RSA_WITH_RC4_128_MD5 • RSA_WITH_RC4_128_SHA
domestic	server	• RSA_WITH_3DES_EDE_CBC_SHA • RSA_WITH_RC4_128_MD5 • RSA_WITH_RC4_128_SHA • RSA_WITH_DES_CBC_SHA • RSA_EX-PORT_WITH_RC4_40_MD5 • RSA_EX-PORT_WITH_DES40_CBC_SHA • TLS_RSA_WITH_NULL_MD5 • TLS_RSA_WITH_NULL_SHA

Profile	Authenticates	Cipher suites
domestic_mutual	client/server	• RSA_WITH_3DES_EDE_CBC_SHA • RSA_WITH_RC4_128_MD5 • RSA_WITH_RC4_128_SHA • RSA_WITH_DES_CBC_SHA • RSA_EXPORT_WITH_RC4_40_MD5 • RSA_EXPORT_WITH_DES40_CBC_SHA • RSA_WITH_NULL_MD5 • RSA_WITH_NULL_SHA

6. Enable the use of IIOPS in the **Communication Ports** subtab by selecting **Secure Management Port** (port 2001). See *Configuring Communication Port Properties* in Sybase Control Center online help.
7. Select the correct security profile name that provides the details for locating the correct certificates.
8. Save the changes and restart the server.
9. Repeat these steps on all remaining servers in the cluster.

Connecting to the Unwired Server Secure Administration Port

Manage the Unwired Platform cluster resource through Sybase Control Center using a secure administration port.

Prerequisites

Enable SSL encryption on each Unwired Server and restart all nodes in the cluster.

Task

1. In the Sybase Control Center Perspective Resources window, right-click the cluster resource and click **Properties**.
2. Change the port number to match the secure management (IIOPS) port. The default is 2001. This enables a secure connection when you log in to the cluster.
3. In the **Use Secure Connection** field, enter `yes`.
4. Click **OK**.
5. Right-click the cluster resource and select **Manage**.

Encrypting Relay Server Connections

Sybase recommends that you encrypt communication to and from a relay server in a production environment.

1. *Reviewing Relay Server Encryption Considerations*

   Understand all relay server production considerations.
2. *Creating a Certificate Authority*

   Create RSA certificates to be used as the certificate authority (CA). Unwired Platform does not support ECC certificates.
3. *Generating a Certificate Request*

   Configure the Web server to use a certificate authority (CA) certificate that you create or another certificate signed by a CA.
4. *Signing the Server Certificate*

   Sign the server certificate request with the createcert utility.
5. *Installing the CA Certificate and Configuring the Server*

   Once you have created and reviewed the certificate, you can safely install it on the server.

Reviewing Relay Server Encryption Considerations

Understand all relay server production considerations.

- Each Relay Server in the production environment requires individual server certificates.
- If using a load balancer the load balancer must also be configured to use SSL. (See your load balancer documentation for details on what may be required.) In this case, Relay Server still needs to be configured to use SSL encryption, otherwise up and down channels between the RSOE and the servers are not encrypted over HTTPS.

Creating a Certificate Authority

Create RSA certificates to be used as the certificate authority (CA). Unwired Platform does not support ECC certificates.

1. At a command prompt, change to `<UnwiredPlatform_InstallDir>UnwiredPlatform\Servers\SQLAnywhere11\BIN32`.
2. Run:

   createcert -s

   The **-s** option generates a root certificate.
3. When prompted, enter `1024` as the **RSA key length**. For all remaining prompts, enter appropriate values for your deployment; for example:

```
<UnwiredPlatform_InstallDir>\UnwiredPlatform\Servers
\SQLAnywhere11\BIN32>createcert
SQL Anywhere X.509 Certificate Generator Version 11.0.1.2405
Enter RSA key length (512-16384): 1024
Generating key pair...
Country Code: US
State/Province: CA
Locality: Dublin
```

```
Organization: MyCompanyCA
Organizational Unit: PTO
Common Name: MyCompanyCA
Enter file path of signer's certificate:
Certificate will be a self-signed root
Serial number [generate GUID]:<enter>
Generated serial number: 3f52ee68c8604e48b8359e0c0128da5a
Certificate valid for how many years (1-100): 10
Certificate Authority (Y/N) [N]: Y
1.  Digital Signature
2.  Nonrepudiation
3.  Key Encipherment
4.  Data Encipherment
5.  Key Agreement
6.  Certificate Signing
7.  CRL Signing
8.  Encipher Only
9.  Decipher Only
Key Usage [6,7]: <enter>
Enter file path to save certificate: rsa_root.crt
Enter file path to save private key: rsa_key.key
Enter password to protect private key: <MyPwd>
Enter file path to save identity: id.pem
```

Note: You must use only an RSA Transport Layer Security (TLS) certificate, and not a certificate generated from ECC TLS or from a third-party source such as **openssl**, or by using a **createcert**- produced certificate from another Sybase product installation.

4. Record your private-key file path and password values, and the certificate and identity file paths. You will need these values again when fulfilling certificate requests for the Web server.

See also

- *Certificate Creation (createcert) Utility* on page 334

Generating a Certificate Request

Configure the Web server to use a certificate authority (CA) certificate that you create or another certificate signed by a CA.

You can use either the server's administration tool or the **createcert** utility to generate the certificate request.

1. To create a signed certificate request on the Web server:
   a. Open your server's administration tool. For example, on IIS, open IIS Manager.
   b. Use the administration tool to create a new certificate request, and save the text file.
2. To create a certificate request with **createcert**:
   a. At a command prompt, change to `<UnwiredPlatform_InstallDir>\UnwiredPlatform\Servers\SQLAnywhere11\BIN32`.
   b. Run:

createcert -r

c. Follow the prompts to complete the required information. For example:

```
<UnwiredPlatform_InstallDir>\UnwiredPlatform\Servers
\SQLAnywhere11\BIN32>createcert -r
SQL Anywhere X.509 Certificate Generator Version 11.0.1.2405
Enter RSA key length (512-16384): 1024
Generating key pair...
Country Code: US
State/Province: CA
Locality: Dublin
Organization: MyCompany, Inc
Organizational Unit: Engineering
Common Name: *.mycompany.com
Enter file path to save request: certreq.txt
Enter file path to save private key: myclient_private.crt
Enter password to protect private key: sybase
```

Note: For the Common Name value in a production environment, Sybase recommends that you use the DNS name of the Unwired Server, or (at minimum) the domain-name where relay server is running.

For a development/test environment that uses SHRS (which runs at `relayserver.sybase.com`), use `*.sybase.com` in Common Name of the certificate. The **.domain-name* is commononly used in a farm of HTTP servers that also uses a load balancer.

Use the generated CA certificate file to sign the server certificate, or do so using the root CA certificate, as described in *Signing the Server Certificate*.

See also

- *Certificate Creation (createcert) Utility* on page 334

Signing the Server Certificate

Sign the server certificate request with the createcert utility.

Prerequisites

Ensure you have created a certificate request either by using the Web server's administration tool or by running `createcert -r`.

Task

1. At a command prompt, change to `<UnwiredPlatform_InstallDir>\UnwiredPlatform\Servers\SQLAnywhere11\BIN32`.
2. Run:

   createcert and use the -s option to point to the certificate request file you created previously.

   For example:

```
createcert -s certreq.txt
```

3. When prompted, enter the root certificate and its private key file and password created earlier to sign this certificate request. For all remaining prompts, enter appropriate values for your deployment; for example:

```
<UnwiredPlatform-InstallDir>\UnwiredPlatform\Servers
\SQLAnywhere11\BIN32>createcert -s certreq.txt
SQL Anywhere X.509 Certificate Generator Version 11.0.1.2405
Enter file path of signer's certificate: rsa_root.crt
Enter file path of signer's private key: rsa_private.crt
Enter password for signer's private key: sybase
Serial number [generate GUID]: <enter>
Generated serial number: ca8295aa17ca41a9a4bd0a3f613c0886
Certificate valid for how many years (1-100): 20
Certificate Authority (Y/N) [N]: N
1.  Digital Signature
2.  Nonrepudiation
3.  Key Encipherment
4.  Data Encipherment
5.  Key Agreement
6.  Certificate Signing
7.  CRL Signing
8.  Encipher Only
9.  Decipher Only
Key Usage [3,4,5]: <enter>
Enter file path to save certificate: myserver.crt
```

   The `myserver.crt` file is installed on the server. Record the file information for future reference.

See also

- *Installing the CA Certificate and Configuring the Server* on page 94

Viewing Certificates

To ensure the certificate was created correctly, run the viewcert utility.

1. At a command prompt, change to `<UnwiredPlatform_InstallDir>\UnwiredPlatform\Servers\SQLAnywhere11\BIN32`.
2. Run:

   viewcert `<myCertFile>.crt`

   In this example, **viewcert** indicates that `rsa_root.crt` is a certificate authority (CA) certificate:

```
<UnwiredPlatform_InstallDir>\UnwiredPlatform\Servers
\SQLAnywhere11\BIN32>viewcert rsa_root.crt
SQL Anywhere X.509 Certificate Viewer Version 11.0.1.2405

X.509 Certificate
-----------------
Common Name:             MyCompany
```

```
Country Code:            US
State/Province:          CA
Locality:                Dublin
Organization:            MyCompany,Inc
Organizational Unit:     PTO
Issuer:                  MyCompany
Serial Number:           c24f9fa714db4f8084d9235dffdf9dcc
Issued:                  May 26, 2010   9:32:00
Expires:                 May 27, 2030   9:32:00
Signature Algorithm:     RSA, SHA1
Key Type:                RSA
Key Size:                1024 bits
Basic Constraints:       Is a certificate authority, path length
limit: 10
Key Usage:               Certificate Signing
```

In this example, **viewcert** indicates that `myserver.crt` is a server certificate:

```
<UnwiredPlatform_InstallDir>\UnwiredPlatform\Servers
\SQLAnywhere11\BIN32>viewcert myserver.crt
SQL Anywhere X.509 Certificate Viewer Version 11.0.1.2405

X.509 Certificate
-----------------
Common Name:             MyCompany
Country Code:            US
State/Province:          CA
Locality:                Dublin
Organization:            MyCompany, Inc
Organizational Unit:     Engineering
Issuer:                  MyCompany
Serial Number:           ca8295aa17ca41a9a4bd0a3f613c0886
Issued:                  May 26, 2010   9:34:00
Expires:                 May 27, 2030   9:34:00
Signature Algorithm:     RSA, SHA1
Key Type:                RSA
Key Size:                1024 bits
Basic Constraints:       Is not a certificate authority
Key Usage:               Key Encipherment, Data Encipherment, Key
Agreement
```

3. Review the values to ensure that the certificate information is correct.

Installing the CA Certificate and Configuring the Server

Once you have created and reviewed the certificate, you can safely install it on the server.

The specific details required for each server type varies. See the documentation that is available with either IIS or Apache. However, we can generalize the steps as follows.

Regardless of the server type, the general steps are:

1. Open your server's administration tool.
   For example, on IIS, open IIS Manager.
2. Process the pending request to install the certificate using the appropriate mechanism.
   Browse to the certificate and select the correct `*.CRT` file.

For example, if you followed the steps described earlier, you are installing `rsa_root.crt`, and `myserver.crt`.

3. Set the SSL port to 443.
4. If you require client certificates, ensure you appropriately configure your Web server. In most cases, client certificates are not required, especially if you are using an LDAP provider.
5. Verify the HTTPS connection in the server by opening a browser and typing `https://<servername>` in the Address bar.
   An Under Construction message indicates that the certificate has been installed correctly.

See also

- *Signing the Server Certificate* on page 92

Encrypting Replication-Based Synchronization Connections

In a production environment for replication-based synchronization (RBS), you must set up both Unwired Server and device clients to use an encrypted port, and manage certificates on both the server and client.

These steps are required to secure communication over the RBS port in an environment where Unwired Server is running in the DMZ or as part of a development environment without a relay server:

1. *Creating Self-Signed Certificates*

   Use createcert to create a self-signed certificate that encrypts replication-based synchronization (RBS) connections. In a production environment, request the certificate from a trusted certificate authority (CA).
2. *Configuring Unwired Server to Use HTTPS for RBS*

   Enable SSL encryption by configuring the replication-based synchronization HTTPS port.
3. *Verifying Device Client HTTPS Setup*

   Complete the encryption of the replication-based synchronization, by verifying the setup of device clients. Typically, device clients are set up by development teams.

Creating Self-Signed Certificates

Use createcert to create a self-signed certificate that encrypts replication-based synchronization (RBS) connections. In a production environment, request the certificate from a trusted certificate authority (CA).

1. At a command prompt, change to `<UnwiredPlatform_InstallDir>\UnwiredPlatform\Servers\SQLAnywhere11\BIN32`.
2. Run:

createcert

3. When prompted, enter `1024` as the **RSA key length**. For all remaining prompts, enter appropriate values for your deployment; for example:

```
<UnwiredPlatform_InstallDir>\UnwiredPlatform\Servers
\SQLAnywhere11\BIN32>createcert
SQL Anywhere X.509 Certificate Generator Version 11.0.1.2405
Enter RSA key length (512-16384): 1024
Generating key pair...
Country Code: US
State/Province: CA
Locality: Dublin
Organization: MyCompanyCA
Organizational Unit: PTO
Common Name: MyCompanyCA
Enter file path of signer's certificate:
Certificate will be a self-signed root
Serial number [generate GUID]:<enter>
Generated serial number: 3f52ee68c8604e48b8359e0c0128da5a
Certificate valid for how many years (1-100): 10
Certificate Authority (Y/N) [N]: Y
1.  Digital Signature
2.  Nonrepudiation
3.  Key Encipherment
4.  Data Encipherment
5.  Key Agreement
6.  Certificate Signing
7.  CRL Signing
8.  Encipher Only
9.  Decipher Only
Key Usage [6,7]: <enter>
Enter file path to save certificate: rsa_root.crt
Enter file path to save private key: rsa_key.key
Enter password to protect private key: <MyPwd>
Enter file path to save identity: id.pem
```

Use the `myserver_identity.crt` file when you configure RBS encryption in Sybase Control Center and use the `mypublic_cert.crt` file when you configure the RBS application that connects to that port. If you are running the RBS server behind a relay server that already uses encryption, these follow-up steps are optional.

Next

Ensure you store your key and identity files in a safe place.

Configuring Unwired Server to Use HTTPS for RBS

Enable SSL encryption by configuring the replication-based synchronization HTTPS port.

1. In the left navigation pane of Sybase Control Center for Unwired Platform, expand the **Servers** node and click the server name.
2. Click **Server Configuration**.

3. In the right administration pane, on the **Replication** tab, click **Synchronization Listener**.
4. Select `Secure synchronization port` as the protocol used for synchronization and configure the certificate properties, then in the optional properties section, specify the `myserver_identity.crt` certificate file using the fully qualified path to the file, along with the password you entered during certificate creation.

   Note: In a clustered environment, this fully qualified path must work for all nodes in the cluster. You can do this via a a shared disk, or distribute this file manually to all nodes.

   See *Configuring Replication-Based Synchronization Properties.* in the Sybase Control Center online help.

Verifying Device Client HTTPS Setup

Complete the encryption of the replication-based synchronization, by verifying the setup of device clients. Typically, device clients are set up by development teams.

See the applicable device platform developer reference for details about how to establish a secure connection with or without relay server.

1. Ensure the application code has been modified to use the HTTPS protocol, port, and stream parameters (with or without a relay server as is appropriate for your environment).
   - If you use relay server, and followed the previous steps in the *Encrypting Replication-Based Synchronization Connections* task workflow, the application developer uses these settings in the code to connect to the relay server's secure port. For example:
     - Port – 443.
     - Protocol – HTTPS (this is equivalent to the MobiLink stream type).
     - Stream parameter –
       ```
       "url_suffix=/ias_relay_server/client/rs_client.dll/
       [SUP_FARM_ID];tls_type=RSA;trusted_certificates=rsa_r
       oot.crt;identity=id_client.pem;identity_password=pwd;
       "
       ```

       Note: The `identity=id_client.pem;identity_password=pwd` segments of the stream parameter are only required if you use a relay server HTTPS port (requires client certificate mutual authentication). This configuration allows the relay server to block denial-of-service attacks at the periphery of you network, should you require that degree of security.
   - If you use Unwired Server in the DMZ or in a development environment without a relay server, and followed the previous steps in the *Encrypting Replication-Based Synchronization Connections* task workflow, then the application developer uses these settings in the code to connect the Unwired Server secure port directly:
     - Unwired Server Port – 2481.
     - Protocol – HTTPS (this is equivalent to the MobiLink Stream Type).

- Stream Parameter – `trusted_certificates=mypublic_cert.crt`

2. Make the `rsa_root.crt` and `id_client.pem` available for the application on the device. They can be included in the application or deployed separately.

Encrypting Messaging-Based Synchronization Connections

Messaging-based synchronization requires no manual setup.

Even though the messages are exchanged using HTTP, messages are fully secured because messaging-based synchronization uses a combination of symmetric (1024-bit) and asymmetric key encryption (128-bit AES) to ensure data privacy.

Encryption keys are based and rotated on a per message basis, not a per session basis. A session is the communication required to transmit item such as MBO data, which has multiple individual messages, each encrypted with its own encryption key. This method ensures key rotation and avoids key stagnation challenges.

Encrypting DCN Connections

Developers are responsible for the construction of data change notification (DCN) requests transmitted from the enterprise information server (EIS). An administrator must enable and configure the HTTPS port for DCN connections.

To enable and configure secure DCN connections to the Unwired Server, administrators must configure a security profile.

1. In Sybase Control Center for Unwired Platform, expand the server node in the corresponding cluster.
2. In the left navigation pane, click **Server Configuration**.
3. In the right administration pane, click **General**.
4. In the **Communication Ports** subtab, expand **Hide secure data change notification ports**, and:
   a) Optionally enter a new port number. The default is 8001.
   b) Enable the desired secure ports.

      If the port value in the **Status** column is **disabled**, click the field and select **Enabled**.
   c) Enter a maximum thread count for each port.

      The value must be a positive integer. The default is 30.
   d) Select an appropriate security profile for the port by clicking the corresponding profile name in the **SSL Security Profile** column, and selecting a profile.
5. If you plan to use a new certificate for this port, create a new certificate, optionally sign it, import it into the Unwired Server keystore (see *Configuring Unwired Server Administration Certificates*, then use the new 104 Sybase Unwired Platform certificate when creating the security profile. Then, assign the security profile to the appropriate DCN ports.

6. If necessary, increase the **Max Thread** default property value.
7. Save the changes and restart the server.
8. Repeat on all particpating servers in the cluster

See also

- *Configuring Unwired Server Administration Certificates* on page 85
- *Data Change Notifications* on page 195

Configuring a Mutually Authenticated SSL Security Configuration

Add and configure a new SSL security profile from Sybase Control Center to use for mutually authenticated connections.

Prerequisites

Your SAP system must be configured for HTTPS mutual authentication.

Task

This procedure is not part of single sign-on (SSO) configuration, instead, it provides an example of how to mutually authenticate DOE and DOE-C connections with an X.509 certificate over HTTPS.

1. From Sybase Control Center, expand **Servers**. Expand the host being configured. Select **Server Configuration**. Select the **General** tab, then the **SSL Configuration** option.
2. Select **<ADD NEW SECURITY PROFILE>** and name it, for example `doectech`. Select `DOECTECH` as the **Certificate alias**, and select **strong_mutual** for **Authentication**.

   This example assumes you have imported the certificate, in this case DOECTECH, into the Unwired Server key store location.
3. Click **Save**, and restart Unwired Server services.
4. Modify the endpoint security profile. For example, a DOE-C endpoint in a domain named E2EDomain with a SAP connection pool named sap_crm:1.0:
   a) Expand **Domains > E2EDomain**, and select **Connections**. Select the **sap_crm:1.0** connection pool.
   b) Click **Properties**, and change the **Endpoint Security Profile** to `doectech`.
   c) Stop Unwired Server services.
   d) Edit the package properties file, and add a new key for `endpoint-security-profile`. Set this to the name of the configured SSL security profile configured in step two. For example:

      ```
      endpoint-security-profile=doectech
      ```

      If defining a security profile to implement mutual authentication with SSO2 or basic authorization, add the optional endpoint property, `techuser-certificate-alias`, which, when set, overrides the technical user name and password fields. The

specified certificate will be extracted from the Unwired Server keystore and supplied to the DOE.

Encrypting Afaria Server Connections for Devices

Enable Afaria device client encryption to protect data that can be transferred between Afaria Servers and remote clients. The security settings are defined, modified and enforced on the client devices using the standard Afaria Security Manager channel. If channel encryption is insufficient, Afaria also allows administrators to encrypt the entire disk as well.

1. *Enabling and Configuring Administration Encryption for Afaria*

   Configure Afaria to communicate with clients over SSL.
2. *Generating an Afaria Certificate*

   Create a new certificate to encrypt only Afaria communications.
3. *Sharing an Unwired Server Certificate with Afaria Server*

   Use the same certificate for both Afaria and Unwired Server SSL encryption.

Enabling and Configuring Administration Encryption for Afaria

Configure Afaria to communicate with clients over SSL.

Prerequisites

Ensure you have Afaria installed and configured properly.

Task

1. Open Afaria Administrator.
2. Configure the correct combination of client communication properties. For details, see:
   - *Afaria Platform Reference Guide > Server Configuration*
   - *Afaria Administrator online help>Properties>Client Communication*

Generating an Afaria Certificate

Create a new certificate to encrypt only Afaria communications.

1. Run the Certificate Creation wizard.
   a) In Afaria Administrator, click **Server List** from the menu at the top of the right pane.
   b) Under the Roles column, click **Administrator**.
   c) On the **Home/Server Status** page, click **Server configuration** from the menu at the top of the right pane.
   d) In the left pane, click **Client communication**.
   e) In the Certificate Creation area, click **Generate certificate request**.

2. To create your certificate request and private key, follow the instructions in the wizard and click **Finish**.
3. Submit the request to a certificate authority (CA) to receive your certificate.

For more information on the Certificate Creation wizard, click the Help link at the top of the Afaria Administrator window and open the *Server Configuration* documentation.

See also

- *Certificate Creation (createcert) Utility* on page 334

Sharing an Unwired Server Certificate with Afaria Server

Use the same certificate for both Afaria and Unwired Server SSL encryption.

You can share a certificate only when the Afaria Server and Unwired Server are installed on the same host, unless you create wildcard certificates (for example, where you specify the domain name of the certificate to be *.<domain>). Wildcard certificates may not always be accepted by clients.

1. Use the **createcert** utility to generate the Unwired Server certificate request.

   Note: If Afaria requires a different format, use a public-domain utility that converts certificates among different formats.
2. Submit the request to a certificate authority (CA) to receive your certificate.

See also

- *Certificate Creation (createcert) Utility* on page 334

User Security Setup

User security ensures that enterprise resources are properly secured when back-end corporate data is mobilized to a device client. Login controls check individual identities, and restrict access to resources on the Unwired Platform-enabled system.

The login security and access control features in Unwired Server include several elements that secure data access from device clients. A security provider is an important part of user security. You can configure one or more security providers within a single security configuration. Creating multiple security configurations allow administrators and domain administrators to scope security stringency according to need for domains, packages, or operations.

A provider uses its own repository, which is a source of information about the users, security roles, security policies, and credentials.

Authentication

A security provider verifies the identities of application users and administrators who request access.

The two types of authentication (users versus administrators) are configured differently:

- For device users – authenticated with custom Unwired Server security configurations created by the platform administrator in Sybase Control Center. A default provider (OpenDS LDAP) is installed with Developer Edition and is suitable for development purposes. However, in a production environment, an enterprise-level security provider must be configured. See *Security for Device Users* later in this chapter.
- For administrators – authenticated with the "admin" security configuration that already exists in Sybase Control Center by default. See *Security for Administration Users* later in this chapter.

See also

- *LDAP Configuration Properties* on page 294
- *NTProxy Configuration Properties* on page 302

Authentication Cache Timeouts

Unwired Server can cache authentication credentials for users in order to improve runtime performance by setting a cache timeout value. These properties used to avoid repeatedly reauthenticating users — a benefit of particular interest for MBS clients, where each message is separately authenticated.

For example, if a user logs in successfully, then during the configured interval, the same user can re-authenticate using the same credentials without validating them against a security repository. However, if the user provides a user name or password that is different from the ones cached, Unwired Server then delegates the authentication request to the security repository.

This property only affects authentication results; authorization results are not cached with this property. By default, the cache timeout value is 3600 minutes (or 3 hours). This value is enabled whether the property exists or not. You can change this value by configuring a new value for the property in Sybase Control Center for the security configuration of your choosing. For details, see *Configuring a Security Configuration* in the Sybase Control Center online help.

Note: If SAPSSOTokenLoginModule is configured as the security configuration, you must also update the "token expiration interval" property . The cache timeout property must be less than the "token expiration interval" property value. See *SAP SSO Token Authentication Properties*.

Single Sign-on Authentication

Single sign-on (SSO) provides an alternative to user name and password authentication, for example X.509 certificate authentication. SSO allows users to enter their credentials only

once to gain access to all resources, including servers, packages, and data sources related to that application.

Authorization

An access policy rule describes which user identity or role is authorized to access a specific resource under named or configured conditions. Unwired Server can be configured to include an authorization provider to authorize device users to access sensitive enterprise data. Sybase Control Center can also authorize administrators to access the complete set of Unwired Platform administration functionality, but domain administrators to only domain specific parts.

Unwired Server authorizes devices users with an authentication provider defined in a new security configuration configured by the platform administrator in Sybase Control Center. By contrast, Sybase Control Center authorizes administrators with the "admin" security configuration in the "default" domain.

Security providers only act as policy enforcers, and therefore do not allow administrators or domain administrators to directly:

- Define, modify, grant, revoke, or delete policies in the enterprise security repository. Only the security administrator has that privilege.
- Manage roles in the enterprise security repository.

  If you require roles, the physical role must exist in the repository. Administrators then later map the logical role used in the mobile business object to this physical role equivalent.

Instead, you must use the third-party administration tools available exclusively for that purpose.

Attribution

In Unwired Platform attribution is typically provided with an LDAP attribution module. The attribution provider has maximum functionality when combined with the LDAP authentication provider, however the attribution provider can be configured completely standalone or with alternate authentication providers.

An attribution provider provides access to standard user attributes such as username, first name, email address, telephone number, as well as any additional attributes. For LDAP, you can configure attribution using the `com.sybase.security.ldap.LDAPAttributer` module.

The attribution provider may provide services in several ways:

- When a user is authenticated through the LDAP authentication provider, the attribution provider is used to fill out more information about the authenticated user.
- If an authenticated client requests information about a specific alternate username, the attribution provider is used to fulfill this request for information using the authenticated user's connection to the LDAP server.

- In the event that a context was not authenticated against an LDAP server, the attribution provider can provide services for client requests for attributes. This works by establishing a new LDAP connection using the configured authentication credentials (BindDN and BindPassword).
- Facilitates self registration. One can create users in the LDAP server using the updated LDAP attributer. When using LDAP attributer to create users in Active Directory, one has to set the configuration property SecurityProtocol to "ssl" and specify the corresponding port as Active Directory requires the use of an SSL connection to set the password attribute for the user.
- (LDAP only) Facilitates changing an expired password for a user who has been validated by the LDAP authentication provider. The user should have failed the authentication attempt with a failure code set to FAILURE_CODE_PASSWORD_EXPIRED_CAN_CHANGE and the same context should be used for changing the expired password as outlined in the SDK.
- Provides a means to query the capabilities supported by the LDAP attributer. The self registration, password change and expired password change capabilities are supported. Password change capability is supported only when the user is authenticated by the LDAP authentication provider. Similarly, the expired password change capability is supported only when the LDAP authentication provider returns the authentication failure code as FAILURE_CODE_PASSWORD_EXPIRED_CAN_CHANGE. When the attributer cannot dynamically determine if the configured credentials have the necessary permissions to create a user or change the password, it supports the capabilities only if the configuration flag AllowSelfRegistrationAndManagement is set to true.

Audit

You can audit security activity by configuring an auditing provider for Unwired Server and Sybase Control Center. The audit provider tracks events such as logins, logouts, server start operations, and so on, including all actions performed by a specific user or with a particular role.

Sybase suggests that administrators audit two types of user activity: successful use of sensitive information and failed attempts at accessing unauthorized information.

Note: On a properly configured and tuned server with adequate capacity, enabling auditing should have minimal impact on the overall system performance.

Security Configurations

A security configuration determines the scope of user identity, authentication and authorization checks, and can be assigned to one or more domains for both messaging and replication-based applications. For example, user "John" may be authenticated one way in one security configuration, but authenticated differently another security configuration, even when "John" represent the same human user.

A security configuration aggregates various security mechanisms for protecting Unwired Platform resources under a specific name, which administrators can then assign to domains. Each security configuration consists of:

- A set of configured security providers
- Security roles
- Role mappings (set at the domain and package level)

A user entry must be stored in the security repository used by the configured security provider to access any resources (that is, either a Sybase Control Center administration feature or an application package to access a data set from a back-end data source). When a user attempts to access a particular resource, Unwired Server tries to authenticate and authorize the user by checking the:

- security policy on the requested resource
- role memberships

Creating and Applying a Security Configuration

Administering security configurations requires an SUP administrator role. Domain administrators cannot modify the assigned security configuration; instead, they can map roles for the configuration at the domain and package level.

1. A platform administrator (that is the, a user with the "SUP Administrator" logical role in the "admin" security configuration):
   a) Creates a security configuration.
   b) Creates and configures the providers used by the security configuration. You must configure at least one authentication provider, however, you can use multiple providers of different types.
   c) Assigns the security configuration to the domain. You must assign at least one security configuration.
   d) Assigns domain administration privileges to users.
2. Once assigned a domain and a security configuration, the domain administrator can set the security configurations for the packages in the domain. If the packages use any logical roles, the domain administrator can also perform role mapping between the logical roles and physical roles of the package security provider.

Stacking Multiple Security Providers

Stack multiple providers to provide a security solution that meets more complex security requirements.

Each provider has a controlFlag attribute that controls overall behavior when you enable two or more providers.

1. Use Sybase Control Center to create a security configuration and add multiple providers as required for authentication, authorization, attribution, and audit. For details, see *Sybase Control Center > Configure > Configure Unwired Platform > Security*.
2. For each provider:
   a) Select the provider name.
   b) Click **Properties**.
   c) Configure the controlFlag property with one of the available values: required, requisite, sufficient, optional.

      See *controlFlag Attribute Values* for descriptions of each available value.
   d) Configure any other common security properties as required.
3. Click **Save**.
4. Select the **General** tab, and click **Apply**.

Stacking LoginModules in SSO Configurations

Use loginmodule stacking to enable role-based authorization for MBOs and data change notification (DCN).

Stacking LDAPLoginModule, LDAPAttributer, and SAPSSOTokenLoginModule to enable role-based authorization with SSO

Neither the SAPSSOTokenLoginModule or the CertificateAuthenticationLoginModule login modules extract role information. If MBOs and MBO operations have roles assigned, stack login modules to get roles for the user, using one of these methods:

1. If SAP is configured to use LDAP/Active Directory as JAAS providers within its Java stack for granting an SSO2 token, configure a stacked LDAPLoginModule pointing to the same LDAP/Active Directory to separately authenticate and retrieve roles. This method assumes the user name and password credentials are authenticated by those modules as well.
2. Rely on the "csi-userrole" provider.

See *Configuring an LDAP Authentication Module* in Sybase Control Center online help.

Stacking modules to enable DCN with SSO

Stack multiple LoginModules with an appropriate set of controlFlag settings to enable DCN in the same SSO enabled package. All DCN operations require the "DCNUser" logical role in the named security configuration (role mapping applies). An additional LoginModule with authorization is required that can assign a physical role. The stacking of modules authenticates DCN users, and grants them the DCN role. Ordering of modules and control flag settings in the security configuration can vary. For example:

1. The SAPSSOTokenLoginModule is first in the list with the control flag set to "sufficient". If authentication succeeds, none of the other Login Modules are called unless their control flags are set to "Required".

2. A Login Module for DCN users that is also "sufficient", and is paired with a module that retrieves roles.

See *Stacking Multiple Security Providers* .

Security Provider Issues

If you experience problems with security configurations or the authentication or authorization providers in these configurations, you need to check the Unwired Server logs for issues.

If you are finding that no errors are being reported, despite failures that may occur while authenticating or authorizing users, you may need to increase the severity level of your logs. To increase severity levels, see *Configuring Server Log Settings* in Sybase Control Center online help.

See also

- *System Logs* on page 246

Built-in Security Providers

Unwired Server supports a variety of built-in security providers. Administrators define one or more security providers when they create a security configuration using Sybase Control Center.

No Security Provider

A NoSec provider offers pass-through security for Unwired Server, which is intended for use in development environments or for deployments that require no security control.

If you use NoSec providers, all login attempts succeed, no matter what values are used for the user name and password. Additionally, all role and control checks based on attributes also succeed.

Sybase has used these classes to implement the NoSec provider:

- **NoSecLoginModule** – provides pass-through authentication services.
- **NoSecAttributer** – provides pass-through attribution services.
- **NoSecAuthorizer** – provides pass-through authorization services.

LDAP Security Provider

The LDAP security provider includes authentication, attribution, and authorization providers.

You can configure these providers:

- The **LDAPLoginModule** provides authentication services. Through appropriate configuration, you can enable certificate authentication in LDAPLoginModule.
- Optional. The **LDAPAuthorizer** or **RoleCheckAuthorizer** provide authorization services for LDAPLoginModule. LDAPLoginModule works with either authorizer. In most production deployments, you must always configure your own authorizer. However, if you

are authenticating against a service other than LDAP, but want to perform authorization against LDAP, you can use the LDAPAuthorizer.

The RoleCheckAuthorizer is always used with every security configuration; however it is not displayed in Sybase Control Center.

Only use LDAPAuthorizer when LDAPLoginModule is not used to perform authentication, but roles are still required to perform authorization checks against the LDAP data store. If you use LDAPAuthorizer, always configure properties for it explicitly. It cannot share the configuration options specified for the LDAPLoginModule (if any are configured).

- Optional. The **LDAPAttributer** provides attribution services. The attribution provider can share configuration defined for the LDAPLoginModule. That is, if no explicit configuration properties are specified for LDAPAttributer, it uses the configuration information from the LDAPLoginModule, when only one LDAPLoginModule is configured. If there are more than one, then LDAPAttibuters cannot share properties, because they would not know which LoginModule to share with; in this case you must also configure propertes for each LDAPAttributer.

You need not enable all LDAP providers. You can also implement some LDAP providers with providers of other types. If you are stacking multiple LDAP providers be sure you understand the configuration implications; see *LDAP Module Stacking Considerations*.

See also

- *LDAP Configuration Properties* on page 294

Active Directory Considerations

If you are using Active Directory as a security provider, ensure you understand the implications of using this LDAP directory in a production environment of Unwired Platform.

Consider these design implications when you extend Active Directory security to Unwired Platform production environments:

- Shared identities among Sybase components – if you are using Active Directory for all authentication requests (that is, administration logins to access Sybase Control Center and application logins to access data), you can also set up Sybase Control Center to use this Active Directory repository. This allows users to have a shared user identity in both components, but Sybase cautions users that this can complicate deployment of the patform. For example, the user identity must be already configured both as an Unwired Platform user as well as an administrator.
- LDAP data structure and administrative control – Unwired Platform requires that roles be added to the repository in specific locations, and therefore, that security configurations also correctly point to the location of the roles. Otherwise, authentication may be problematic. Coordinate the implementation of these roles with the LDAP administrator.

See also

- *LDAP Configuration Properties* on page 294

Multiple LDAP Trees for Authentication and Authorization

Use the Unwired Platform administration perspective to configure the LDAP authentication and authorization security providers, which are used to locate LDAP user information when organizational user groups exist within multiple LDAP trees.

You can use these approaches to accommodate an LDAP tree structure that cannot be directly accessed using one search base:

- Create an LDAP authentication module for each level in the hierarchy – during the authentication process, Unwired Platform tries to authenticate against every login module in the ordered list until authentication succeeds or until it reaches the end of the list. Depending on the number of login modules you configure, this approach may have some performance issues.
- Use different AuthenticationScopes for performing user searches – specify the root node of a particular LDAP tree, by entering `AuthenticationSearchBase="dc=sybase, dc=com"` and `set Scope=subtree`. Unwired Platform performs an LDAP query against the entire subtree for authentication and authorization information. Depending on the number of AuthenticationScope within the LDAP tree structure, this approach can have performance implications.
- Implement a proprietary login module – create a custom logic and search mechanism by implementing the login module callback interface. This approach involves Java coding to a set of public interfaces within Unwired Platform.

LDAP Role Computation

Role checks are the primary means of performing access control when using LDAP authentication. Both the authentication and attribution capabilities utilize role computation techniques to enumerate the list of roles that both authenticated and non-authenticated users have.

There are three distinct types of role constructs supported by LDAP providers; each may be used independently or all three may be configured to be used at the same time.

- User-level role attributes, which are specified by the UserRoleMembershipAttributes configuration property, are arguably the most desirable and efficient role definition format. Using this technique, a user's roles are enumerated by a read-only directory server-managed attribute on the user's LDAP record. The advantage to this technique is efficiency in which role memberships can be queried and the ease of management using the native LDAP server's management tools. These constructs are supported directly by ActiveDirectory products. The configuration options necessary for this technique are:
  - UserRoleMembershipAttributes – the multi-valued attribute on the user's LDAP record that lists the role DN's that the user is a member of. An example value for this property is "memberOf" on ActiveDirectory.

- RoleSearchBase – the search base under which all user roles are found. Examples of this are "ou=Roles,dc=sybase,dc=com", for example. This value may be the root search base of the directory server as well.
- RoleFilter – the search filter that, coupled with the search base, retrieves all roles on the server.
- RoleScope – This value may optionally be specified to enable retrieval of roles from subcontexts under the search base.
- RoleNameAttribute – This value may optionally be specified to choose an attribute other than "cn" to define the name of roles.

These properties will retrieve correct values automatically based on the type of server you configure, despite having subtly different schemas.

- LDAP Group role definitions may be used as role definitions. This is a common construct among older LDAP servers, but is supported by nearly all LDAP servers. This may be a useful technique when wanting to use the same LDAP schema across multiple LDAP server types. Unlike the user-level role attributes, LDAP group memberships are stored and checked on a group-by-group basis. Each defined group, typically of objectclass "groupofnames" or "groupofuniquenames", has an attribute listing all of the members of the group. The configuration settings used are the same as for user-level role attributes, except for the RoleMemberAttributes property which replaces the UserRoleMembershipAttributes property. This property defines a comma-delimited list of attributes that contain the members of the group. An example value for this property is uniquemember,member, which represents the membership attributes in the above-mentioned LDAP objectclasses.
- Freeform role definitions are unique in that the role itself does not have an actual entry in the LDAP database. A freeform role starts with the definition of one or more user-level attributes. When roles are calculated for a user, the collective values of the attributes (which can be multi-valued) are added as roles the user is a member of. This technique is particularly useful when the overhead of managing roles uses are administration-heavy. One interesting example would be assigning a freeform role definition that was equivalent to the department # of the user. A role check performed on a specific department number would only be satisfied by users who have the appropriate department # attribute value. The only property that is required or used for this role mapping technique is the comma-delimited UserFreeformRoleMembershipAttributes property.

LDAP Module Stacking Considerations

LDAP providers can sometimes share a common configuration.

If you want to share the configuration between the login provider and the attribution provider, set the configuration for the login provider but do not specify values for the attribution provider. In this case, the attribution provider will automatically inherit the appropriate configuration from the login provider.

Note: You cannot share configuration between the login provider and the authorization provider. The authorization provider always expects its own configuration.

Other considerations include:

- The attribution provider has maximum functionality when combined with the LDAP authentication provider; however, the attribution provider can be configured completely standalone or with alternate authentication providers.
- If you do not configure a login provider, you must define the configure all properties in the attribution provider instead.
- A complete configuration defined for the attribution provider override that of a login module. While it is possible to have separate configurations for the authentication and attribution providers, Sybase does not recommend this setup.

NTProxy Security Provider

NTProxy (sometimes known as native Windows login) integrates with existing Windows login security. Users can authenticate with their native Windows user name and password, which gives them access to roles that are based on their existing Windows memberships.

The NTProxy provider fulfills authentication services only with classes in `csi-nativeos.jar`; role-based access control, and attribution are not directly supported. Groups are also not supported in NTProxy. Instead, group memberships are transformed into a role of the same name and can be mapped in Sybase Control Center. See *Sybase Control Center > Configure > Configuring Unwired Platform > Role Mapping Configuration*.

See also

- *NTProxy Configuration Properties* on page 302
- *Setting Up the NTProxy Provider* on page 118

SAPSSOTokenLoginModule Authentication Provider

Use the Unwired Server SAPSSOTokenLoginModule authentication provider to implement SSO with an SAP enterprise information system (EIS) using SSO2 Tokens.

Use this authentication provider for both JCo and DOE-C connections to the SAP system. Unwired Server does not provide authorization control or role mappings for user authorization. Instead, enforce any access control policies in the SAP System.

Before assigning authentication providers, you must download and install SAP cryptographic libraries and SSO2Token files from the SAP Marketplace.

See also

- *SAP SSO Token Authentication Properties* on page 306
- *Installing the SAP SSO2Token Files on Unwired Server Hosts* on page 135

CertificateAuthenticationLoginModule Authentication Provider

Use the Unwired Server CertificateAuthenticationLoginModule authentication provider to implement SSO with an enterprise information system (EIS) with X.509 certificates.

Unwired Server does not provide authorization control or role mappings for user authorization. Instead, enforce any access control policies in the EIS.

Before assigning authentication providers, you must download and install the appropriate libraries from the EIS. For example, the SAP cryptographic libraries from the SAP Marketplace.

See also

- *Certificate Authentication Properties* on page 303
- *Installing the SAP Cryptographic Libraries on Unwired Platform* on page 34
- *Setting up SAP SSO Using X.509 Certificates* on page 129

Roles and Mappings

Role mapping occurs when an administrator maps logical roles to physical roles using Sybase Control Center as part of a security configuration or a deployment package. The physical roles are the roles and groups in the underlying security repository. The mapped role determines the security role requirement for a user at runtime to access a resource that is using the security configuration on which the mapping is defined.

In Unwired Platform, the mapped role determines what security roles apply to users when they attempt to perform an operation from the mobile application (device users) or Sybase Control Center (administrators).

Role mappings are defined as part of a security configuration that you can assign to a particular domain. Administrators can assign the same security configuration to multiple domains; ensure that these mappings are suitable for all domains to which the security configuration is assigned. Consider an example where security configuration is shared between domainA and domainB.

1. The platform administrator (the administrator assigned the SUP administration role) creates a security configuration called AllDomains.
2. The platform administrator assigns the AllDomain to the domain, and maps the EmpRole role to SalesGroupRole in the security repository used by that configuration.

This change that is specific to just domainA is also implemented in domainB even though the domain administrator of domainB did not explicitly make, or require, the change. But the role mapping is propagated to domainB as well. To avoid this, the Unwired Platform administrator may want to create multiple security configurations so that underlying mechanisms can stay the same, but specific role mappings can be made for each.

For device user security, there is an increased flexibility for packages as they are deployed. If a security configuration is inappropriate, or if a role is not mapped at all that is used by the package, the platform or domain administrator can override or extend the role mappings

defined for the security configuration. Package-level role mappings always take precedence in such a scenario.

See also

- *Mapping Roles for a Domain* on page 161
- *Mapping Roles for a Package* on page 183

Physical Roles

Physical roles are named references to roles or groups that an administrator has defined on a back-end enterprise security provider. Mapping a logical role to a physical role allows authorization control in the Unwired Server. Replicate these names exactly, so that logical roles can be mapped correctly to the server role.

Logical Roles

Default logical roles are administration roles already setup in Unwired Platform. Other roles will typically be defined by package developers, and allow developers to define identities that each application uses to indicate access rights to its different objects. Logical roles may or may not replicate physical names already defined for your security provider.

Logical roles allow secured access to Unwired Platform resources for several users at once, and you can define them for one or more packages in a domain. If a developer has created a logical role for a package, you may need to manually map it to one or more physical roles.

In the absence of explicit mapping, the default role mapping is set to AUTO, which is equivalent of logical role mapping to a physical role of the same name, in the underlying provider of that security configuration.

The following default roles are used: SUP Administrator, SUP Domain Administrator, and SUP DCN User. These roles must be mapped. In a development environment, mappings are automatically created. In a production environment, physical roles must be added manually to the directory used, and then manually mapped in Sybase Control Center.

Mapping State Reference

The mapping state determines the authorization behavior for a logical name instance.

State	Description
AUTO	Map the logical role to a physical role of the same name. Both the logical role and the physical role must match, otherwise, authorization fails.
NONE	Disable the logical role, which means that the logical role is not authorized. This mapping state prohibits anyone from accessing the resource (MBO or Operation). Use this option after carefully considering potential consequences.

State	Description
MAPPED	A state that is applied after you have actively mapped the logical role to one or more physical roles. Click the cell adjacent to the logical role name and scroll to the bottom of the list to see the list of mapped physical roles.

Dynamically Mapping Physical Roles to Logical Roles

Map roles at either a domain or package-level, depending on the scope requirements of a particular binding. If you use a particular role mapping for a package and a different role mapping at the domain-level, the package mapping overrides the domain-level mapping.

In Sybase Control Center for Unwired Platform, determine where the role mapping needs to be applied:

- For domain-level mappings, configure role mappings as part of the security configuration for a domain.
- For package-level mappings, configure role mappings when you deploy a package to Unwired Server, or at the package-level after deployment.

SUP DCN User Role

The SUP DCN User is a logical role that Unwired Platform uses to authorize any DCN event: updating data in the cache, executing an operation, or triggering a workflow package.

Before any DCN event is submitted, the person or group mapped to this role must be authenticated and authorized by the security configuration used. By default, SUP DCN User is automatically available to all new security configurations you create. However, the underlying default varies depending on the environment in use.

- In a development environment – A physical role called DCNRole is automatically added to the OpenDS LDAP directory. The SUP DCN User logical role is mapped to the DCNRole physical role, which is group automatically added to this directory. Both the supAdmin and supDcnDeveloper LDAP users are added as members of DCNRole, so either user may perform DCN.
- In a production environment – In a single domain environment, the "default" Domain's security configuration has "no security" set by default. That means, any user/password credentials are authenticated, and all roles are granted to everyone. So any user could perform DCN initially.

  However, eventually this default configuration for the default domain will change. In this case—as in the case for any additional security configurations that is added—the SUP DCN User logical role must be mapped to some physical role in the backend security systems, and the user who performs DCN must be in that physical role.

  Note: If security configuration provider does not support roles, you must perform a special mapping manually. You can create a single user role (a role explicitly mapped to a single user authenticated against the particular security configuration). This is achieved by prefixing the username with `user:`. For example, a mapped physical role named

`user:joep` would authorize the user named 'joep' to issue DCN to any package associated with the particular security configuration that contains this mapping, or to issue a workflow DCN to any user authenticated against the particular security configuration.

To map the SUP DCN User to a user in the underlying security repository, the user name must be first defined in Sybase Control Center as a physical role that is mapable. Then, SUP DCN User role can be mapped to a physical user or to a physical role from Sybase Control Center. For example, if you want to map SUP DCN User to a user use the format `user:<User>` . Alternatively, you can also map it to a role with `<PhysicalRole>`.

If you are supporting multiple domains, then the user name also needs to include the named security configuration for the package the DCN is targeted for, by appending *@<DomainSecurityConfigName>* as a suffix to that name. Suppose you have two packages (PKG_A, PKG_B) deployed to 2 domains (Domain_A, Domain_B) respectively. Further, assume that PKG_A in Domain_A has been assigned to the "admin" security configuration, whereas PKG_B in Domain_B has been assigned to the "alternateSecurityConfig" security configuration.

- A user doing DCN to PKG_A should identify themselves as *User@admin*.
- A user doing DCN to PKG_B should identify themselves as *User@alternateSecurityConfig*.

If you are using ActiveDirectory, and are using email addresses for user names, then definitions appear as *<username@myaddress>@<DomainSecurityConfigName>*.

Furthermore, the implementation varies depending on the DCN service used:

- For workflows, because the resource the user is pushing data towards is a group of named users (users authenticated previously successfully against a certain security configuration), therefore the user must have the authorization to push to that particular security configuration. The user must have be mapped to SUP DCN User in the security configuration for the workflow target.
- A user having SUP DCN User logical role in security configuration 'mySecConfig1' must not have the right to push workflow DCN or regular DCN to a user or package associated with 'mySecConfig2'.

Security for Administration Users

Sybase Control Center requires its own set of security providers, which are distinct from those you must set up for Unwired Platform security. Typically, the only providers an administrator would configure for Sybase Control Center would be authentication and authorization.

1. *Configuring a Security Provider for Sybase Control Center*

   In a production environment, once you have added required users to the repository used for Sybase Control Center authentication, you can use that directory to authenticate administration login requests. In a development environment, changing the security provider of "No Security" is optional.
2. *Setting Up Provider Roles in Sybase Control Center*

Configure the security provider administration and user roles and passwords required for Sybase Control Center administrator login.

3. *Setting Up the Admin Security Configuration*

   Configure the admin security configuration to authenticate all administrator users. In a development environment, the admin security configuration points to the OpenDS LDAP server and the default role mapping. To secure your Unwired Server infrastructure in the development environment, you must use Sybase Control Center to set the admin security configuration to point to your security repository server and configure the appropriate role mapping.

4. *Mapping and Assigning Unwired Platform Default Roles*

   Unwired Platform requires that you map these default platform roles: SUP Administrator, SUP Domain Administrator, and SUP DCN User. In a development environment, these mappings are automatic. In a production environment, use Sybase Control Center to map these logical roles to the appropriate physical roles or groups in the underlying security provider.

Configuring a Security Provider for Sybase Control Center

In a production environment, once you have added required users to the repository used for Sybase Control Center authentication, you can use that directory to authenticate administration login requests. In a development environment, changing the security provider of "No Security" is optional.

Prerequisites

Backup the `<SCC_HOME>\conf\csi.properties`.

Task

1. Exit Sybase Control Center.
2. From Windows Services panel, stop the Sybase Unified Agent Service.
3. From a text editor, open `<SCC_HOME>\conf\csi.properties`.
4. Define a module in this file, similar to the LDAP sample below.

   Each line of the LDAP server module of the properties file must begin with "CSI.loginModule." followed by a module number. The module number in this sample is 8. The module number must be unique in the properties file, and you must use the same number in every line of the module.

   For example, this module configures an LDAP provider module using Active Directory, so that administrators can log in to Sybase Control Center with their Windows user name.

```
================================================

CSI.loginModule.
8.options.AuthenticationSearchBase=ou=sup,dc=mycompany,dc=com
CSI.loginModule.
8.options.BindDN=CN=suppad,ou=sup,dc=mycompany,dc=com
```

```
CSI.loginModule.8.options.BindPassword=mybindpassword
CSI.loginModule.
8.options.DefaultSearchBase=ou=sup,dc=mycompany,dc=com
CSI.loginModule.
8.options.AuthenticationFilter=(&(sAMAccountName={uid})
(objectclass=user))
CSI.loginModule.8.options.RoleFilter=(&(objectclass=groupofnames)
(objectclass=group))
CSI.loginModule.8.options.RoleScope=subtree
CSI.loginModule.8.options.AuthenticationScope=subtree
CSI.loginModule.8.options.ProviderURL=ldap://msadserver:389
CSI.loginModule.
8.options.RoleSearchBase=ou=sup,dc=mycompany,dc=com
CSI.loginModule.8.options.ServerType=msad2k
CSI.loginModule.8.options.moduleName=SUP LDAP Login Module
CSI.loginModule.8.controlFlag=sufficient
CSI.loginModule.
8.provider=com.sybase.ua.services.security.ldap.LDAPWithRoleLogin
Module
CSI.loginModule.8.debug=false
=====================================================
```

Note: Change the values for only lines shown in bold.

For a complete list of available LDAP properties you can configure, see *System Administration > System Reference > Security Provider Configuration Properties > LDAP Configuration Properties*.

5. For some internal communication, you must include the Anonymous Login Module in the `csi.properties` file:

   ```
   # Anonymous Login Module
   CSI.loginModule.
   0.provider=com.sybase.ua.services.security.anonymous.AnonymousLog
   inModule
   CSI.loginModule.0.controlFlag=sufficient
   CSI.loginModule.0.options.moduleName=Anonymous Login Module
   CSI.loginModule.0.options.roles=uaAnonymous
   ```

   Adding this anonymous login module does not relax or allow anonymous access to the Sybase Control Center. Authentication and authorization checks are still enforced.

6. Save the file.
7. If your LDAP server's SSL certificate is signed by a nonstandard certificate authority (for example, if it is self-signed), use the **keytool** utility to configure JVM to trust the certificate. Execute a command similar to this:

   ```
   keytool -import -keystore <SUP_installdir>\shared\JRE-<version>
   \bin\keytool\lib\security\cacerts -file
   <your cert file and path> -alias ldapcert -storepass changeit
   ```

8. Restart Sybase Unified Agent.
9. Open Sybase Control Center and log in.

See also

- *Setting Up Provider Roles in Sybase Control Center* on page 119

Setting Up the NTProxy Provider

To use a Windows NTProxy login provider, set up the Windows account as well as an `admin` security configuration in Sybase Control Center.

This security provider is typically used only to configure administrator security in Sybase Control Center.

1. Add Unwired Platform groups and users to Windows by running the **Control Panel** and opening **User Accounts**.
   - Create a SUP Administrator and a SUP Domain Administrator group.
   - Add administrator users and passwords to that group. For example, the default user and password is supAdmin and s3pAdmin respectively.
2. Configure NTProxy for Sybase Control Center, by opening `SCC_HOME\conf\csi.properties`, and adding an NTProxy login module. For example:

   ```
   CSI.loginModule.
   5.provider=com.sybase.ua.services.security.os.NTProxyLoginModule
   CSI.loginModule.5.controlFlag=sufficient
   CSI.loginModule.5.options.moduleName=NT Proxy Login Module
   CSI.loginModule.5.options.debug=true
   ```

3. Manually map roles for Sybase Control Center, by opening `SCC_HOME\conf\roles-map.xml`, and editing the `NTProxy Login Module` entity to add SUP Administrator and SUP Domain Administrator group mappings.

   The "name" property must match the "moduleName" property value in csi.properties.

   Note: The value in "modRole" are Windows OS groups designated for Unwired Platform administrator and Domain administrator.

   This example uses "SUP Administrator" as the Windows OS group for platform administrator and "SUP Domain Administrator" for domain administrator rights. Make sure to create those groups or use your group name, and assign users membership to respective group as needed:

   ```
   <module name="NT Proxy Login Module">
         <role-mapping modRole="Administrators"
   uafRole="uaAnonymous,uaAgentAdmin" />
         <role-mapping modRole="Administrators"
   uafRole="uaAnonymous,uaOSAdmin" />
        <role-mapping modRole="Users" uafRole="uaAnonymous,uaUser" /
   >
         <role-mapping modRole="Guests"
   uafRole="uaAnonymous,uaGuest" />
         <role-mapping modRole="sybase"
   uafRole="uaAnonymous,uaPluginAdmin,sccUserRole" />
         <role-mapping modRole="SUP Administrator"
   uafRole="uaAnonymous,uaAgentAdmin,uaPluginAdmin,sccAdminRole,sccU
   serRole" />
   ```

```
                        <role-mapping modRole="SUP Domain
Administrator"
uafRole="uaAnonymous,uaAgentAdmin,uaPluginAdmin,sccUserRole" />
</module>
```

You can also map roles in Sybase Control Center. See *Assigning a Role to a Login or a Group* in the Sybase Control Center online help.

4. Configure the NTProxy provider for the "admin" security configuration in SCC.
   a) Log in as a SUP administrator.

      Supply the domain as well as the user name, for example: `userName@domainName`
   b) In the left navigation tree, click **Security** > **admin** to modify the existing security configuration used for SCC.
   c) In the right administration pane, select the **Authentication** tab and click **New**.
   d) Select **NTProxy** as the authentication provider.
      - If you are using the local domain, ensure that the **Default Authentication Server** property value is the name of this host.
      - If you are using a Windows domain, set the value to the name of the Windows domain authentication server host. You do not need to set the property; not setting the property enables auto-detection of the authentication server.

      See *NTProxy Configuration Properties* for property descriptions.
   e) Switch to the **Attribution** tab, and make sure to add an LDAP Attributer provider, if it is not already listed, and remove any other attributer.
   f) Configure other properties as required and click **OK**.
   g) Select the Unwired Server name, and click **Apply** to save the changes to the `admin` security configuration.

The next time you log in to SCC, use your Windows user account name and password to authenticate as an administrator or domain administrator. A domain name is still required. For example: instead of supAdmin/s3pAdmin, use **userName@domainName**/WindowsPwd.

See also

- *NTProxy Security Provider* on page 111
- *NTProxy Configuration Properties* on page 302

Setting Up Provider Roles in Sybase Control Center

Configure the security provider administration and user roles and passwords required for Sybase Control Center administrator login.

1. Exit Sybase Control Center.
2. From Windows Services panel, stop the Sybase Unified Agent Service.
3. Ensure that the roles defined in the LDAP repository match the roles defined in the `<SCC_HOME>\conf\roles-map.xml` file.

By default, the role mapping file contains these roles in the LDAP Login Module definition:

```
 <module name="SUP LDAP Login Module">
      <role-mapping modRole="SUP Administrator"
uafRole="uaAnonymous,uaAgentAdmin,uaPluginAdmin,sccAdminRole,sccU
serRole" />
      <role-mapping modRole="SUP Domain Administrator"
uafRole="uaAnonymous,uaAgentAdmin,uaPluginAdmin,sccAdminRole,sccU
serRole" />
    </module>
```

You can change "SUP Administrator" and "SUP Domain Administrator" to reflect the role names that you want to use for administrator and domain administrator authentication.

4. To add role mapping for the Anonymous Login Module to the uaAnonymous role:
   a) Add the uaAnonymous role to the `<uaf-roles>` section of the `roles-map.xml` file:
      ```
      <role name="uaAnonymous" description="Anonymous role" />
      ```
   b) Add the role mapping in the `<security-modules>` section of the `roles-map.xml` file:
      ```
      <module name=''Anonymous Login Module''>
                        <role-mapping modRole=''uaAnonymous''
      uafRole=''uaAnonymous'' />
          </module>
      ```
5. Restart Sybase Unified Agent.
6. Open Sybase Control Center and log in.

See also

- *Configuring a Security Provider for Sybase Control Center* on page 116

Setting Up the Admin Security Configuration

Configure the `admin` security configuration to authenticate all administrator users. In a development environment, the `admin` security configuration points to the OpenDS LDAP server and the default role mapping. To secure your Unwired Server infrastructure in the development environment, you must use Sybase Control Center to set the `admin` security configuration to point to your security repository server and configure the appropriate role mapping.

1. Open Sybase Control Center.
2. In the left navigation pane, expand the **Security** folder.
3. Select **Admin**.
4. In the right administration pane, configure authentication and authorization by clicking the corresponding tab, and configuring properties of the providers you add as required.

By default, authentication and authorization uses No Security. You can customize the security configuration by removing this provider and other types of providers.

5. Configure the same LDAP login provider that is used to authenticate and authorize Sybase Control Center administrative users. You can use a single provider can be used for both Sybase Control Center and Unwired Platform administrators.
   a) Click **New**.
   b) Select the provider.
   c) Set the controlFlag property to sufficient. Then ensure that all property values that are suitable for your security requirements.
   d) Set this provider to come before the NoSecLoginModule.

      Note: Do not delete this module yet. Otherwise, if you have misconfigured the new login module, you may have issues logging in to Sybase Control Center to correct the issue.

   e) Click **OK**, to save the configuration locally.
6. Ensure the login has, and ordered before the NoSecLoginModule on the list.
7. Click **Save** to save the security configuration changes.
8. Test the security before removing the NoSecLoginModule from the security configuration.
   a) Set security logging to a DEBUG value.
   b) Logout of Sybase Control Center and login with a user ID from the new security backend who should have SUP Administrator role.
   c) Check the server log to see if the new login module authenticated the user successfully. If not, modify the login provider properties until you can login successfully.
   d) Once the new login module is working correctly, remove the NoSecLoginModule providers and save the configuration.
   e) Set security logging to INFO, or something more appropriate.

Mapping and Assigning Unwired Platform Default Roles

Unwired Platform requires that you map these default platform roles: SUP Administrator, SUP Domain Administrator, and SUP DCN User. In a development environment, these mappings are automatic. In a production environment, use Sybase Control Center to map these logical roles to the appropriate physical roles or groups in the underlying security provider.

The mapping of these logical roles to one or more physical roles in the underlying security provider decides whether a platform or domain administrator has access privileges. The administration logical to physical role mapping is done in 'default' domain for the 'admin' security configuration at the domain-level.

1. Open Sybase Control Center.
2. In the left navigation pane, expand **Domains**.

3. Expand the **Default** domain.
4. Open the Security folder and click the **Admin** security configuration.
5. Map roles to the security provider groups or roles:
   - If default roles are exactly match the names used in the security provider repository, select **AUTO**.
   - If the default role differs from those manually added to the repository, click the list adjacent to the logical role and choose **Map Role**. This command displays the Role Mappings dialog, which allows you to manually set the logical and physical role mappings. The dialog shows the name of the logical role you are mapping in the text area of the dialog. Once saved, the state automatically changes to MAPPED.

   See *Sybase Control Center > Configure > Configure Unwired Platform > Domains > Configuring Domain Security*.
6. Assign domain administration access to users in Sybase Control Center:
   a) Register the user by clicking the **Security** node and selecting the **Domain Administrators** tab, then clicking **New**.
   b) Assign the required domain administrator physical role to the user in the underlying security provider repository for **admin** security configuration.

   See *Sybase Control Center > Configure > Configure Unwired Platform > Domains > Administrators*.
7. Restart Unwired Server.

Security for Device Users

Unwired Server requires administrators to configure security provider for device users. The production environment for your device applications may require you to create multiple security configurations of different types of providers.

Use Sybase Control Center to configure security providers for device users security.

For example, a company sales employee needs to look up a client's phone number in a phone book device application. This authentication sequence allows the sales employee to access data from the phone book application:

1. The employee tries to open the application, which prompts for a user name and password, which is local to the device, and not explicitly tied to a corporate security account.
2. The first time the application is opened, the employee must synchronize the customer MBO to access the client phone number.
3. Unwired Server gets an authentication request.
4. Unwired Server sends the request to the authentication provider that processes the login credentials.
5. The provider checks the user name and password against information stored in authentication repository, in this case, an LDAP directory server on the corporate LAN.

6. The directory server evaluates the access policy to see if the authenticated user has permission to access this client's contact information.
7. If the login request is valid, the user is authenticated. Because the employee has the correct access privileges, Unwired Server is notified and the resource request is fulfilled.

   Note: The granularity of access control checks is at the MBO-class or MBO-operation level. Therefore, if the user has access to one customer record, he or she can access all customer records.
8. If the login request is invalid, an error is generated and authentication fails.

For example, a user needs to access a client application locally, without a connection to the Unwired Server. The application uses the offline authentication method from the package database class to authenticate against the last successfully authenticated credentials.

1. The user tries to open the application, which prompts for a user name and password, which is local to the device, and not explicitly tied to a corporate security account.
2. The first time the application is opened, the offline authentication method verifies with the client database if those credentials are valid.
3. If the username and password are correct, the method returns successfully and the user is authenticated. The user accesses the client application locally.
4. If the login request is invalid, an error is generated and authorization fails.
5. If the credentials are invalid, an error is generated and authentication fails.
6. If the user doesn't have a required role, an error is generated and synchronization fails.

Single Sign-on for SAP

Several single-sign on access policies are available for the SAP environment. This enables users to be authenticated using a SAP logon ticket (SS02 Ticket) or an X.509 certificate.

Both replication- and message-based mobile clients can use SSO with:

- **SAP data source support –**
  - SAP Data Orchestration Engine Connector (DOE-C) connections
  - SAP Java Connector (JCo) connections
  - SAP BAPIs exposed as Web services – both HTTP and HTTPS endpoints are supported

  For information about developing mobile business objects for SSO, see the topic *Implementing SSO for SAP* in Unwired WorkSpace online help.
- **Single sign-on credential support –**
  - X.509 certificates – at runtime, the mobile client selects the certificate signed by a trusted CA, which is authenticated by the SAP EIS.
  - SAP single sign-on (SSO2) tokens – at runtime, the client enters a user name/password combination that maps to a user name/password in the SAP EIS. A token is obtained from the configured SAP server using the client supplied user name/password and is

forwarded to other SAP servers configured in the endpoints to authenticate the client, instead of using client supplied user name/password credentials.

- **Client support –**
  - Windows Mobile, iOS, and BlackBerry message-based clients
  - Prebuilt applications, such as the Sybase Mobile Sales for SAP CRM
  - Windows Mobile, iOS, and BlackBerry replication-based clients
  - Workflow clients

See also

- *Certificate Authentication Properties* on page 303
- *Installing the SAP Cryptographic Libraries on Unwired Platform* on page 34
- *Setting up SAP SSO Using X.509 Certificates* on page 129

Implementing SSO for SAP

Follow these procedures to implement SSO with an SAP enterprise information system (EIS). Procedures vary depending on your specific scenario, for example SSO2 versus X.509 credentials, and involves both server and application configuration.

See also

- *Preparing Unwired Server Hosts for SSO* on page 126
- *Preparing Your SAP Environment for SSO* on page 128
- *Setting up SAP SSO Using X.509 Certificates* on page 129
- *Setting up SAP SSO Using SSO2 Tokens* on page 134

Server Configuration for SSO

Sybase Control Center online help and the System Administration Guide contains information about setting up SAP and Unwired Servers for SSO, and establishing connections between Unwired Server hosted applications and packages and SAP enterprise information systems.

Goal	Task	Procedures required to achieve the goals
Verify the SAP EIS is configured correctly to accept SSO connections from Unwired Server.	Preparing your SAP environment	*Preparing Unwired Server Hosts for SSO* – required for all scenarios

Goal	Task	Procedures required to achieve the goals
Install the required SAP software on Unwired Server hosts.	Preparing your Unwired Server hosts	• *Installing SAP Cryptographic Libraries on Unwired Server Hosts* – required for all scenarios • *Installing SAP Java Connectors (JCo) on Unwired Server Hosts* – required for scenarios that use a JCo connection • *Installing DOE-C on Unwired Server* – required for scenarios that use a DOE-C connection • *Installing the SAP SSO2Token Files on Unwired Server Hosts* – required for scenarios that use SSO2 credentials
Configure and assign a security configuration that implements either SSO2 or X.509 credentials to packages or domains.	Configuring Unwired Server	• *Configuring an SAP JCo Connection to an SAP Server* – required for scenarios that use a JCo connection • *Creating and Assigning a Security Configuration That Uses SSO2 Tokens*– required for scenarios that use SSO2 credentials • *Creating and Assigning a Security Configuration That Uses X.509 Credentials*– required for scenarios that use X.509 credentials

MBO Development for SSO

Unwired WorkSpace online help contains details about how to configure MBOs so they can be used in SSO implementations

Goals	Procedures required to achieve the goals
Configure your MBOs so they use username and password personalization keys and bind them to data sources to which those credentials are propagated.	• *Creating an SAP Connection Profile* and *SAP Java Connector (JCo) Prerequisites* – required for all scenarios that use an SAP MBO • *Propagating a Client's Credentials to the Back-end Data Source* – required for all scenarios • *Configuring an MBO to Use an SAP Java Connector*– required for scenarios that use SAP MBOs and a JCo connection • *Creating a Web Service Connection Profile* and *Configuring an SAP Exposed Web Service MBO to Use Credentials*– required for scenarios that use BAPIs exposed as Web services

Application Configuration for SSO

The type of application and credentials for which you are configuring SSO determines the location of the configuration details.

Application type	Goals	Procedures required to achieve the goals
Prepackaged applications that use DOE-C, such as CRM, and ESDMA bundles	Modify SAP endpoints in ESDMA bundles to https://host:8001/endpoint. Add the technical user and DOE-C endpoint security profile certificate to the DOE.	• *Configuring a Mutually Authenticated SSL Security Configuration* • *SAP DOE-C Properties* In the *System Administration Guide* – required for scenarios that use DOE-C
Workflow applications	Add and configure a credential starting point for the workflow application.	• *Installing and Testing X.509 Certificates on Simulators and Mobile Devices*– required for scenarios that use X.509 credentials • *Configuring the Workflow Application to Use Credentials* – required for scenarios that use SSO2 or X.509 credentials In the *Developer Guide for Mobile Workflow Packages*
Replication-based applications	Install and import X.509 certificates and use the object API to select them for client connections.	• *Installing and Testing X.509 Certificates on Simulators and Mobile Devices* • *Single Sign-On With X.509 Certificate Related Object API* In the Developer Guides for iOS, Windows and Windows Mobile, and BlackBerry

Preparing Unwired Server Hosts for SSO

Before configuring Unwired Server to use SSO, import and install required libraries and modules on Unwired Server hosts.

Prerequisites

Before you can access the SAP Web site, you must have an SAP account.

Task

These steps describe the common prerequisite tasks for setting up SAP to use cryptographic libraries in the Unwired Server runtime. Sybase recommends that you make all changes

concurrently in a distributed environment for development and production installations of Unwired Platform.

1. *Configuring an SAP JCo Connection to an SAP Server*

   Create a Java Connection (JCo) to an SAP Server in Unwired Server from Sybase Control Center.
2. *Deploying Packages and Bundles to Domains*

   Before assigning a security configuration to the domain and package that are to use SSO, deploy the package or bundle to an Unwired Server domain .

See also

- *Implementing SSO for SAP* on page 124
- *Preparing Your SAP Environment for SSO* on page 128
- *Setting up SAP SSO Using X.509 Certificates* on page 129
- *Setting up SAP SSO Using SSO2 Tokens* on page 134

Configuring an SAP JCo Connection to an SAP Server

Create a Java Connection (JCo) to an SAP Server in Unwired Server from Sybase Control Center.

Prerequisites

Start Unwired Server services and log in to Sybase Control Center as the administrator.

Task

The SAP JCo connection provides access for various client types, including those that use SSO2 tokens and X.509 certificates.

1. Expand the cluster, expand the **Domains** folder, expand the domain to which the package is to be deployed, and select **Connections**.
2. Select the **Connections** tab and click **New**. Name the connection pool `SAP Server`, select **SAP** as the Connection pool type, select the **sap template**, and enter appropriate properties for your environment. For example:
   - Language (jco.client.lang) = `EN`
   - Host name (jco.client.ashost) = `sap-doe-vm1.sybase.com`
   - System number (jco.client.sysnr) = `00`
   - SNC mode (jco.client.snc_mode) = `1`
   - SNC name (jco.snc_myname) = `p:CN=SNCTEST, O=Sybase, L=Dublin, SP=California, C=US`
   - SNC service library path (jco.client.snc_lib) = `C:/sapcryptolib/sapcrypto.dll` (the location of the cryptographic library)

- Client number (jco.client.client) = `100`
- SNC partner (jco.client.snc_partnername) = `p:CN=sap-doe-vm1, OU=SUP, O=Sybase, C=US`
- SNC level (jco.client.snc_qop) = `1`

3. Click **Test Connection** to verify access to the SAP server, and click **OK**.

Deploying Packages and Bundles to Domains

Before assigning a security configuration to the domain and package that are to use SSO, deploy the package or bundle to an Unwired Server domain .

1. Deploy the package/bundle/BAPI exposed as a Web Service, and so on from Unwired WorkSpace to Unwired Server.
2. During deployment, select a domain (default, for example).
3. Configure the package/bundle as required.
   - **ESDMA bundle –** Edit the properties file to include `login-required=true`. Restart Unwired Server.

     If you skip this step, the ESDMA bundle supports only basic authentication.
   - **SAP_CRM application –** Edit the properties file to include `login-required=true`, and set the `endpoint-security-profile` as described in the topic *Configuring a Mutually Authenticated SSL Security Configuration* for mutually authenticated sessions.

Preparing Your SAP Environment for SSO

Verify the SAP EIS is configured correctly to accept SSO connections from Unwired Server.

The general steps for enabling SAP systems to communicate over HTTPS are:

1. Install the SAP Cryptographic library.
2. Set all credential parameters for the type of credentials accepted by the server:
   - SSO2 token – verify everything is set properly with SSO2 transaction.
   - X.509 certificate – set up, import, and verify the certificates using the trust manager (transaction STRUST).
3. Use the ICM configuration utility to enable the ICM HTTPS port.
4. Set the type of authentication to enable over HTTPS:
   - Server authentication only – the server does not expect the client to authenticate itself using SSL, only basic authentication
   - Client authentication only – the server requires the client to send authentication information only via SSL certificates. The ABAP stack supports both options. Configure the server to use SSL with client authentication by setting the parameter (ICM/HTTPS/verify_client):
     - 0 – do not use certificates.

    - 1 – allow certificates (default).
    - 2 – require certificates.
5. Use the trust manager (transaction STRUST) for each PSE (SSL server PSE and SSL client PSE) to make the server's digitally signed public-key certificates available. Use a public key infrastructure (PKI) to get the certificates signed and in the SAP system.

   There are no SSO access restrictions for MBO data that spans multiple SAP servers.
6. Unwired Server must possess a valid CA X.509 certificate exported from SAP. Deploy these certificates, which are used during SSL handshake with the SAP server into the Unwired Server trust store.
7. The user identification (distinguished name), specified in the certificate must map to a valid user ID in the AS ABAP, which is maintained by the SM30 view (VUSREXTID).

See *Configuring the AS ABAP for Supporting SSL* at *http://help.sap.com/saphelp_aii710/helpdata/en/49/23501ebf5a1902e10000000a42189c/frameset.htm*

See also

- *SAP SSO Token Authentication Properties* on page 306
- *Installing the SAP SSO2Token Files on Unwired Server Hosts* on page 135
- *Certificate Authentication Properties* on page 303
- *Installing the SAP Cryptographic Libraries on Unwired Platform* on page 34
- *Setting up SAP SSO Using X.509 Certificates* on page 129
- *Implementing SSO for SAP* on page 124
- *Preparing Unwired Server Hosts for SSO* on page 126
- *Setting up SAP SSO Using SSO2 Tokens* on page 134

Setting up SAP SSO Using X.509 Certificates

Follow these procedures to define a trusted relationship between Unwired Server and the SAP EIS using X.509 credentials, and assign a security configuration to a package or domain that supports X.509 authentication.

1. *Generating and Installing an X.509 Test Certificate on Unwired Server*

   Generate an X.509 certificate on Unwired Server to use in testing SSO connections with SAP Systems.
2. *Creating a Keystore and Importing an X.509 Certificate and Private Key*

   Create a keystore on the Unwired Server host into which you can import the certificate and private key (PKCS12) issued by the SAP system administrator, which is required for SSO using X.509 certificates.
3. *Installing and Configuring Certificates on Unwired Server*

   Use the Java keytool command to import SAP certificates into the Unwired Server truststore.

4. *Implementing SSO in an Unwired Server Cluster*

   Place required files in the appropriate primary Unwired Server subdirectory so they are distributed to all Unwired Servers within the cluster during cluster synchronization.

5. *Creating and Assigning a Security Configuration That Uses X.509 Credentials*

   Create a new security configuration, assign the CertificateAuthenticationLoginModule authentication provider to it, and assign the security configuration to an Unwired Server domain.

See also

- *CertificateAuthenticationLoginModule Authentication Provider* on page 112
- *Single Sign-on for SAP* on page 123
- *Preparing Your SAP Environment for SSO* on page 128
- *Certificate Authentication Properties* on page 303
- *Implementing SSO for SAP* on page 124
- *Preparing Unwired Server Hosts for SSO* on page 126
- *Setting up SAP SSO Using SSO2 Tokens* on page 134

Generating and Installing an X.509 Test Certificate on Unwired Server

Generate an X.509 certificate on Unwired Server to use in testing SSO connections with SAP Systems.

These instructions describe how to generate an X.509 certificate for testing SAP and SSO only. In a production environment, a different entity controls certificate management. For example, an SAP system administrator controls certificate generation and management for his or her particular environment, including maintaining the certificate list in a Personal Security Environment (PSE) with trust manager.

Note: When the CertificateAuthenticationLoginModule gets a certificate from a client, it can optionally validate that it is a trusted certificate. The easiest way to support validation is to import the CA certificate into the `<UnwiredPlatform_InstallDir>/Servers/UnwiredServer/Repository/Security/truststore.jks` file, which is the default Unwired Server truststore.

Use the SAPGENPSE utility to create a PSE certificate to use for testing. See *http://help.sap.com/saphelp_nw04s/helpdata/en/a6/f19a3dc0d82453e10000000a114084/content.htm*. The basic steps are:

1. Generate the certificate from the `C:\sso\sapcryptolib` directory:

   ```
   sapgenpse get_pse <additional_options> -p <PSE_Name> –r
   <cert_req_file_name> -x <PIN> <Distinguished_Name>
   ```

2. Copy the PSE certificate (for example, SNCTEST.pse) to the location of your installed SAP cryptographic libraries. For example, `C:\sapcryptolib`.

3. Generate a credential file (cred_v2) from the `C:\sapcryptolib` directory:

   ```
   sapgenpse seclogin -p SNCTEST.pse -O DOMAIN\your_name_here
   -x password
   ```

   Note: The user generating the certificate must have the same user name as the process (mlserv32.dll or eclipse.exe) under which the Unwired Platform service runs.

Creating a Keystore and Importing an X.509 Certificate and Private Key

Create a keystore on the Unwired Server host into which you can import the certificate and private key (PKCS12) issued by the SAP system administrator, which is required for SSO using X.509 certificates.

Create a keystore and add the SAP issued DOETECH certificate and private key to the Unwired Server host. You can do this in a variety of ways. One way is to implement the **Not Yet Commons** library to create a KeyStoreBuilder, described at the *http://juliusdavies.ca/commons-ssl/index.html* Web site. For example:

1. Shut down Unwired Server.
2. Add `@java -cp %~dp0not-yet-commons-ssl-0.3.11.jar org.apache.commons.ssl.KeyStoreBuilder` to the `KeyStoreBuilder.bat` file.
3. Produce a `DOECTECH.jks` Java keystore: `KeyStoreBuilder password DOECTECH.p12`
4. Change the certificate and keystore passwords, and import the certificates from the source keystore to the target, using these **keytool** commands:

   ```
   keytool -storepasswd -new changeit -keystore DOECTECH.jks -
   storepass password
   ```

   ```
   keytool -keypasswd -alias doectech -keypass password -
   storepass changeit -new changeit -keystore DOECTECH.jks
   ```

   ```
   keytool -importkeystore -destkeystore l:\Sybase
   \UnwiredPlatform\Servers\UnwiredServer\Repository\Security
   \keystore.jks -srckeystore DOECTECH.jks
   ```

5. When prompted, enter these responses:

   Enter destination keystore password: `changeit`

   Enter source keystore password: `changeit`

If successful, you see this output:

```
Entry for alias doectech successfully imported.
Import command completed: 1 entries successfully imported, 0 entries
failed or cancelled
```

Installing and Configuring Certificates on Unwired Server

Use the Java keytool command to import SAP certificates into the Unwired Server truststore.

1. Shut down Unwired Server and all services.
2. Import the SAP system's certificate into the Unwired Server truststore:

```
keytool -importcert -keystore l:/Sybase/UnwiredPlatform/Servers/
UnwiredServer/Repository/Security/truststore.jks -file
l:/x/uepmain-638146-7-sso-e2e/certs/sap-doe-vm1.cert
Enter keystore password: changeit
Owner: CN=sap-doe-vm1.sybase.com, OU=I0020159296, O=Sybase, C=US
Issuer: EMAILADDRESS=jdoe@sybase.com, CN=sybase.com, OU=Unwired
Enterprise, O="Sybase, Inc.", L=Dublin, ST=California, C=US
Serial number: 7
Valid from: Thu Oct 14 19:01:02 MDT 2010 until: Fri Oct 14 19:01:02
MDT 2011
Certificate fingerprints:
MD5: 1B:95:8A:AE:56:C5:F0:44:1A:02:AD:AA:10:CB:11:2D
SHA1: 47:53:54:9D:55:0B:39:B8:6A:6E:A6:8A:5C:68:51:7E:DF:8C:EF:F8
Signature algorithm name: SHA1withRSA
Version: 1
Trust this certificate? [no]: y
Certificate was added to keystore
```

Implementing SSO in an Unwired Server Cluster

Place required files in the appropriate primary Unwired Server subdirectory so they are distributed to all Unwired Servers within the cluster during cluster synchronization.

Any changes to a named security configuration affect the cluster and triggers a cluster synchronization, which automatically zips the files in the primary Unwired Server `CSI` subdirectory and distributes them to the other servers in the cluster. Copy all certificate and other security-related files to the `CSI` subdirectory.

SSO2 tokens are cached in the Unwired Server CDB, and since all nodes of the cluster share the primary server's CDB, it is accessible by all other nodes in the cluster.

The provider configuration information, which includes the server certificate file name and location, must be the same on all cluster nodes. The same is true for the cryptographic DLLs and certificate files for SSO using X509.

1. On the primary server in the cluster, put any SAP certificate files or truststores into the `<UnwiredPlatform_InstallDir>\Servers\UnwiredServer\Repository\CSI\conf` directory on the primary server.

   Use system properties to specify the full path and location of the file in the configuration so they can be accessed from different servers within the cluster if installation directories are different from that of the primary server. For example:

```
${djc.home}/Repository/CSI/conf/
SNCTEST.pse
```

For X.509 CertificateAuthenticationLoginModule, if the *ValidateCertificatePath* is set to true, the default, the CA certificate (or one of its parents) must be installed in the trust store for each server.

Note: Unwired Server truststore and keystore files:

- `<UnwiredPlatform_InstallDir\Servers\UnwiredServer \Repository\Security\truststore.jks` – is the Unwired Server trust store that contains CA (or parent) certificates. Unwired Server trusts all CA or parent certificates in `truststore.jks`.
- `<UnwiredPlatform_InstallDir\Servers\UnwiredServer \Repository\Security\keystore.jks` – contains client certificates only.

The CertificateAuthenticationLoginModule also has `Trusted Certificate Store*` and `Store Password` properties which you can set if you want to keep the module out of the default Unwired Server trust store. In which case you must first:

a) Use **keytool** to put the CA certificate into a new keystore.
b) Put the keystore into the `Repository\CSI\conf` subdirectory.
c) Include the path in the `Trusted Certificate Store` property.

2. From Sybase Control Center, add the login module.
3. Restart all Unwired Servers within the cluster.

Creating and Assigning a Security Configuration That Uses X.509 Credentials

Create a new security configuration, assign the CertificateAuthenticationLoginModule authentication provider to it, and assign the security configuration to an Unwired Server domain.

Prerequisites

Install CertificateAuthenticationLoginModule dependencies. See *Installing SAP Cryptographic Libraries on Unwired Server*.

Task

The CertificateAuthenticationLoginModule authentication provider supports X.509 certificate logins to SAP systems through both JCo and DOE-C connections. You can assign security configurations to domains or packages.

1. Log in to Sybase Control Center. Navigate to and select **Security**.
2. Select the **General** tab, then **New**.
3. Name the secrurity configuration, for example `X509SECADMINCERT`, and click **OK**.
4. Select the **X509SECADMINCERT** security configuration.
5. Select the **Authentication** tab.

6. Click **New** and select **com.sybase.security.core.CertificateAuthenticationLoginModule** as the Authentication provider.
7. Click **OK** to accept the default settings, or modify any additional settings as required.
8. Select **com.sybase.security.core.NoSecLoginModule** and click **Delete**.
9. Select the **General** tab, select **Validate**, then **Apply**.
10. Navigate to the domain to which you are assigning the security configuration, and select the **Security Configurations** tab.
11. Click **Assign**, and select **X509SECADMINCERT** to assign the security configuration to the domain.
12. Select the **Security** tab, and remove any other security configurations for the domain, if configured.

Setting up SAP SSO Using SSO2 Tokens

Follow these procedures to define a trusted relationship between Unwired Server and the SAP EIS using SSO2 credentials, and assign a security configuration to a package or domain that supports SSO2 authentication.

1. *Installing the SAP SSO2Token Files on Unwired Server Hosts*

   (Optional) Unzip and install the contents of the latest SSO2Token zip file in all nodes of your Unwired Server cluster. This library is only required if you use single sign-on with SSO2 tokens and want to enable persisted token caching.
2. *Creating and Assigning a Security Configuration That Uses SSO2 Tokens*

   Create a new security configuration, assign the SAPSSOTokenLoginModule authentication provider to it, and assign the security configuration to an Unwired Server domain.

See also

- *Implementing SSO for SAP* on page 124
- *Preparing Unwired Server Hosts for SSO* on page 126
- *Preparing Your SAP Environment for SSO* on page 128
- *Setting up SAP SSO Using X.509 Certificates* on page 129

Installing the SAP SSO2Token Files on Unwired Server Hosts

(Optional) Unzip and install the contents of the latest SSO2Token zip file in all nodes of your Unwired Server cluster. This library is only required if you use single sign-on with SSO2 tokens **and** want to enable persisted token caching.

Prerequisites

Download and install the SAPCAR utility, which is required to extract the contents of the SAR archive.

Task

The installation package is available to authorized customers on the SAP Service Marketplace. There are different SAPSSOEXT packages for various hardware processors.

1. Go to the SAP Web site at *http://service.sap.com/patches*, select **Browse our Download Catalog**.
2. Select **Additional Components** and select **SAPSSOEXT**.
3. Select and download the latest **SAPSSOEXT** library for your platform.
4. Unzip the file using the SAPCAR utility, which contains:
   - `sapsecu.dll`
   - `sapssoext.dll`
5. Copy `sapsecu.dll` to `C:\WINDOWS\system32`, regardless of platform.
6. Copy `sapssoext.dll` to `<UnwiredPlatform_InstallDir>\UnwiredPlatform\Servers\UnwiredServer\lib`, regardless of platform.

See also

- *SAPSSOTokenLoginModule Authentication Provider* on page 111
- *Preparing Your SAP Environment for SSO* on page 128
- *SAP SSO Token Authentication Properties* on page 306

Creating and Assigning a Security Configuration That Uses SSO2 Tokens

Create a new security configuration, assign the SAPSSOTokenLoginModule authentication provider to it, and assign the security configuration to an Unwired Server domain.

The SAPSSOTokenLoginModule authentication provider supports SSO2 token logins to SAP systems through both JCo and DOE-C connections. You can assign security configurations to domains or packages.

1. Log in to Sybase Control Center. Navigate to and select **Security**.
2. Select the **General** tab, then **New**.
3. Name the secrurity configuration, for example `SAPSSOSECADMIN`, and click **OK**.

4. Select the **SAPSSOSECADMIN** security configuration.
5. Select the **Authentication** tab.
6. Click **New** and select **com.sybase.security.sap.SAPSSOTokenLoginModule** as the Authentication provider. Enter SAP server URL, certificate name, and password values appropriate for your SAP system and click **OK**. For example:

```
SAP Server URL: http://sap-doe-vm1.sybase.com:8000/sap/bc/ping?
sap-client=200
SAP Server Certificate: ${djc.home}/Repository/CSI/conf/
SNCTEST.pse
//full path with server-specific environment variable set to root
of imported SAP PSE certificate
SAP Server Certificate Password: password
```

Note: The SAP Server URL should match that of the deployed ESDMA.

7. Select **com.sybase.security.core.NoSecLoginModule** and click **Delete**.
8. Select the **General** tab, select **Validate** then **Apply**.
9. Navigate to the domain to which you are assigning a security configuration, and select the **Security Configurations** tab.
10. Click **Assign**,and select **SAPSSOSECADMIN** to assign the security configuration to the domain.
11. Select the **Security Configurations** tab, and remove any other security configurations for the domain, if configured.

Data Security Setup

Administrators can implement data security in two ways: changing passwords required to access the component databases used in Unwired Platform, and encrypting data in these repositories.

1. *Protecting System Data Access*

   Data stored in cluster and consolidated database can contain varying degrees of sensitive data; Sybase recommends that you immediately change the default DBA password to one that is more secure.
2. *Data Encryption Implementation*

   Encrypt the data used by Unwired Platform to add an additional layer of platform security.

Protecting System Data Access

Data stored in cluster and consolidated database can contain varying degrees of sensitive data; Sybase recommends that you immediately change the default DBA password to one that is more secure.

1. *Changing the Consolidated Database DBA Password*

Run the Interactive SQL utility (dbisql) to change the DBA password.

2. *Changing the CDB Password and Registering Changes Among Components*

   The Platform administrator can update the modified password in Unwired Server configuration files.

3. *Verifying the DSN Entries*

   After you have changed the DBA passwords, verify that the data sources still work.

See also

- *Data Encryption Implementation* on page 139

Changing the Consolidated Database DBA Password

Run the Interactive SQL utility (**dbisql**) to change the DBA password.

Prerequisites

Ensure runtime servers are stopped. In Windows, select **Start > Programs > Sybase > Unwired Platform<version> > Stop Unwired Platform Services**.

Task

1. Change to `<UnwiredPlatform_InstallDir>\Servers\SQLAnywhere11\bin32`, and run:

   ```
   dbisql
   ```

2. Locate the database you are changing the password for:
   a) Select **ODBC data source name**.
   b) Click **Browse**.
   c) Select the database. You may need to select **Show all data sources** to see all names used by Unwired Platform.

      For the consolidated database, choose **default-CDB** if you are using the default SQL Anywhere consolidated database.

   d) Click **OK**.

3. Enter:

   ```
   grant connect to dba identified by <NewPwd>
   ```

4. Execute the command by clicking the right arrow.
5. Exit **dbisql**.
6. Repeat these steps for the cluster database. If you are using the default DSN names, the cluster DSN is clusterdb_<clustername>.

See also

- *Changing the CDB Password and Registering Changes Among Components* on page 138

DBA Passwords

By default, the Unwired Platform creates many component databases, and Unwired Server accesses these databases with the DBA user identity.

For the default consolidated database of SQL Anywhere, a single user ID of DBA is created, and the password is initially sql (passwords are case-sensitive). The DBA user ID automatically gives Unwired Server the DBA authority within these component database, which enables Unwired Server to perform any activity in the database. Unwired Server can create required tables, change table structures, and so on. For this reason, Sybase strongly recommends you change these default passwords to increase the level of protection to Unwired Platform cache and metadata.

Changing the CDB Password and Registering Changes Among Components

The Platform administrator can update the modified password in Unwired Server configuration files.

1. Open `<UnwiredPlatform_InstallDir>\Sybase\UnwiredPlatform\Servers\UnwiredServer\Repository\Instance\com\sybase\sup\server\SUPServer\sup.properties`.
2. Modify the cdb.password and cldb.password properties to set a new password.

   Passwords are entered in clear text password. After you have completed the next step, all passwords are encrypted and the sup.properties file is resaved with the new encrypted values.
3. Run `<UnwiredPlatform_InstallDir>\Sybase\UnwiredPlatform\Servers\UnwiredServer\bin\configure-mms.bat <clustername>`.

   In a single server setup, <clustername> is host name of this computer. In a cluster setup, <clustername> is the hostname of first node installed in the cluster.

See also

- *Changing the Consolidated Database DBA Password* on page 137

Verifying the DSN Entries

After you have changed the DBA passwords, verify that the data sources still work.

1. Select **Control Panel** > **Administrative Tools** > **Data Sources (ODBC)**.
2. Select the **System DSN** tab.
3. Double-click **default-cdb**, then click **Test Connection**.
   Look for a `"Connection Successful"` message.
4. Return to the **System DSN** tab.
5. Double-click **clusterdb_<*cluster name*>**, then click **Test Connection**.

Look for a `"Connection Successful"` message.

Data Encryption Implementation

Encrypt the data used by Unwired Platform to add an additional layer of platform security.

Some Unwired Platform components do not support encryption. Review this table to see which components can enable this security feature.

Component	Encryption available?	Implementation notes
Various embedded databases	No	Not applicable
Device data	Yes, but optional	Use the Data Security Manager with Afaria and encrypt either some of the data or the entire disk. Encrypted data is always decrypted before it is packaged and transported. Therefore, Sybase recommends that if security is a concern, that you encrypt the communication channel used.
Device client database	Yes, but optional	A developer can enable database encryption by using either an API or a database property. A developer can encrypt PIM data or entity application data—in other words, personal and enterprise data that is cached persistently on the device, and is normally accessed by the user via a corresponding Unwired Platform mobile workflow or mobile business object. Encrypted data is always decrypted before it is packaged and transported. Therefore, Sybase recommends that if security is a concern, that you encrypt the communication channel used.

See also

- *Protecting System Data Access* on page 136
- *Chapter 13, Customization with the Client API* on page 283

Encrypting Data on the Windows Device Client

Sybase recommends that you use Afaria to enable data encryption. Afaria data encryption uses a component called Data Security Manager, which is available only with the correct license of Sybase Unwired Platform.

Each Afaria client supports a different level of data encryption.

1. Contact Sybase about obtaining the correct license for device data encryption.
2. With the correct license in place, choose the client type and level of encryption desired.

   Only some device clients are shared by Unwired Server and Afaria:

Encryption level	Device client	Implemented by
Full disk	Windows	Administrator
PIM data	Windows and Windows Mobile	Administrator selects th PIM data that Afaria can encrypt calendar, e-mail, and so on.
Path/folder	Windows and Windows Mobile/Windows Mobile Professional	Administrator defines the encryption target. However, a user can copy files to the encrypted area of a disk.
Path/folder	Windows Mobile Professional	User defines encryption target.

   You can find complete information about configuring data encryption in Afaria documentation. See *Afaria Reference/Components > Data Security Manager for Handheld Clients > Encryption Options* guide for details. For information about which subtypes of Windows Mobile are supported by the Data Security Manager License, see the system requirements documented in the *Afaria Release Notes*.

CHAPTER 7 Server Environment Administration

Server environment administration includes all the activities required to set up and configure your Unwired Platform server environment.

Administration of the server environment includes:

- Managing Unwired Server clusters. The membership (adding or removing a node), naming, and unlinking of a cluster's relay server is all handled during the installation or manual configuration process.

  Versioning of the cluster occurs automatically and cannot be directly controlled by the administrator. If you have a cluster-affecting configuration change performed in Sybase Control Center for the primary server, the change is shared with the rest of the cluster.
- Configuring the server environment from Sybase Control Center.
- Enabling and designing domains for multiple tenants in your system with Sybase Control Center.
- Managing and monitoring your consolidated database. Decide whether to use a new default database or an existing one, and configure the environment to use the corresponding database. You can do this with the installer or with Sybase Control Center postinstallation (for existing databases only).

Cluster Administration Overview

The goal of cluster administration is to ensure that clusters and servers work smoothly, and scale over time. Cluster administration is mostly a nonroutine administration task.

By default, the Unwired Platform is installed as a one-node cluster, which may be sufficient for development or test environments. However, production deployments of Unwired Platform are likely to require multiple nodes.

Table 8. Cluster administration tasks

Task	Frequency	Accomplished by
Installing the cluster	One-time installation per cluster	Unwired Platform installer
Setting up relay servers	One-time initial installation and configuration; occasionally adding servers to the cluster	Manual installation; manual setup using configuration files.

Task	Frequency	Accomplished by
Suspending and resuming server nodes	On demand, as required	Sybase Control Center
Setting cluster properties, including consolidated database settings, monitoring database setup, and so on	Once, or as cluster changes require	Manual configuration using files and `.BAT` scripts.
Administering the runtime databases	Routine to ensure that the database server is monitored and backed up, that there is sufficient space for Unwired Platform metadata and cached data tables, and that performance is within acceptable limits (performance tuning)	Established processes and command line utilities. Consult with your database administrator.
Reviewing licensing information, including total licensed devices and currently used licenses count	Occasional, or as device user registration and deregistration occurs	Sybase Control Center.

See also

- *Server Administration Overview* on page 142
- *Setting Up Unwired Server Nodes* on page 17

Server Administration Overview

The goal of server administration is to ensure that Unwired Server is running correctly and that it is configured correctly for the environment in which it is installed (development or production). Server administration is mostly a one-time or infrequent administration task.

Table 9. Server administration tasks

Task	Frequency	Accomplished by
Installing the server	One-time installation per server	Unwired Platform installer.

Task	Frequency	Accomplished by
Configuring the server to: • Set the replication and messaging synchronization ports, as well as communication ports for administration and DCN • Create security profiles for secure administration communication • Set up secure synchronization • View consolidated database properties • Configure replication and messaging push notifications • Set transport level security properties within a security profile • Tune server performance	Postinstallation configuration with infrequent tuning as required	Sybase Control Center for Unwired Platform. See *Sybase Control Center online help > Configure > Configuring Unwired Platform > Unwired Server*.
Setting server log file settings and subsystem log levels	Once, unless log data requirements change	Sybase Control Center for Unwired Platform. See *Sybase Control Center online help > Configure > Configuring Unwired Platform > Unwired Server > Server Logs*.

See also

- *Cluster Administration Overview* on page 141
- *Setting Up Unwired Server Nodes* on page 17
- *Installing Third-party Software* on page 18

Configuring the IIOP Socket Listener

By default, the IIOP socket listeners is configured to listen on the first IP address resolved for your host name.

To configure Unwired Server to listen on all network interfaces, edit each listener's properties file.

1. Stop Unwired Server.
2. On the system where Sybase Unwired Platform is installed, navigate to the `<UnwiredPlatform_InstallDir>\UnwiredPlatform\Servers\UnwiredServer\Repository\Instance\com\sybase\djc\server\SocketListener` directory.
3. Use a text editor to open the `<hostname>_iiop1.properties` file.
4. Locate the line beginning with `host=`:

```
#Instance Properties
#Mon Feb 22 10:34:27 PST 2010
```

```
maxThreads=200
port=2000
host=<your_host_name>
ant.project=default-socket-listeners
useSocketChannel=false
protocol=iiop
```

5. Replace your host name with `0.0.0.0`:

```
#Instance Properties
#Mon Feb 22 10:34:27 PST 2010
maxThreads=200
port=2000
host=0.0.0.0
ant.project=default-socket-listeners
useSocketChannel=false
protocol=iiop
```

6. You may also need to tune the performance of the listener by editing the default maxThreads value. If you change the messaging queue count properties, then you may need to set a new value where the maxThread of iiop socket listener is larger than the sum of all nodes needed iiop thread counts.

   Sybase recommends the following:

   - In a single node environment with a messaging server, use this equation: `outbound queue count + 50`. This is because each outbound workflow will occupy an IIOP thread, so the maximum thread value must be greater than the number occupied in the outbound queue.
   - In a clustered environment , use this equation for the primary server:`(outbound queue count + 50 + ((inbound queue count + outbound queue count + 50) * number of secondary nodes)` . In this case, each inbound and outbound workflow on secondary nodes also occupies an IIOP thread on the primary server. So assuming the cluster has multiple secondary node, each primary server has to multiply the value of the inbound and outbound queue to get an sufficiently high value.

     For secondary servers in the cluster, default value is sufficient.

7. Save and close the properties file.
8. Restart Unwired Server.

See also

- *Configuring Unwired Server Performance Properties* on page 149
- *Synchronization Model Performance Tuning* on page 37

Viewing Consolidated Database Properties

Review the consolidated database (CDB) properties that allow Unwired Server to connect to the database.

You cannot use Sybase Control Center to configure CDB properties. .

1. Open Sybase Control Center.
2. In the left navigation pane, expand the **Servers** folder and select a server.
3. Select **Server Configuration**.
4. In the right administration pane, click the **Consolidated DB** tab.
5. Review these properties:
   - **Database Thread Count** – the number of worker threads used for the CDB. The default value is 20 threads. However, if you are experiencing performance issues, especially in a clustered environment, you may need to increase this value.

     For a SQL Anywhere CDB only, use this formula to estimate a new value:

     ```
     Value = Number of nodes in cluster * (sync threadcount +
     1) + Number of scheduled EIS fetches + 10
     ```

     For example, if you have been using the default synchronization thread count of 20, but have added three Unwired Servers to your cluster, adjust the CDB thread count to 78 or (3 * (20 +1) + 5 + 10). If you set the value for this thread count to 78 or higher, the value is accepted. However, if you set the value lower than 78, the request is ignored, unless you remove some servers from the cluster or reduce the synchronization thread counts.
   - **Database Type** – the CDB type; Sybase_ASA for the default SQL Anywhere database.
   - **Database DSN Name** – CDB DSN name descriptor.
   - **Database Name** – database name descriptor.
   - **Database Server Port** – the port over which CDB communication takes place. The default is 5200.
   - **Database Password** – the database user password. In SQL Anywhere, the default is `sql`.
   - **Database Server Host** – the name of the machine where the existing database server is running.
   - **Database Server** – the name of the database server used to manage requests for the consolidated database. By default, the server is localhost. However, if the database is on another host or is part of a cluster, you may need to use another host name.
   - **Database User** – the CDB user name. In SQL Anywhere, the default user name is `dba`.
6. (Optional) Expand the **Show Optional Properties** section and review values for these properties:
   - **Database Install Type** – the type of database installation; either default or custom.
   - **Database ASA Mode** – the database mode. The value of the first node of a cluster is "primary," the value of the second node is "arbiter," and the value of the third node is "mirror."
   - **Database User Options** – specify the CDB start-up user options.

Configuring a Synchronization Listener for Replication-Based Synchronization

Configure the port to receive replication-based synchronization requests from client devices, and if you are using push synchronization, then also configure synchronization listener properties.

Prerequisites

Determine whether you require an encrypted (secured) or unencrypted synchronization stream. A secure synchronization stream uses SSL encryption; therefore, before setting up a secure configuration, ensure that you possess digital certificates verified and signed by third-party trusted authorities. The HTTPS protocol is slower than the HTTP protocol; use SSL only if you require HTTPS. See *Transport Security Setup*.

Task

1. Open Sybase Control Center.
2. In the left navigation pane, expand the **Servers** folder and select the server you want to configure.
3. Select **Server Configuration**.
4. In the right administration pane, click the **Replication** tab.
5. If push synchronization is being added to replication-based synchronization application, select **Synchronization Listener** from the menu bar.
6. Select the protocol and port you require:
   - If you do not require SSL encryption, choose **Synchronization port**. Sybase recommends this option if you do not require a secure communication stream for synchronization. By default, the port for HTTP is 2480.
   - To encrypt the HTTP stream with SSL, choose **Secure Synchronization port**. By default, the port for HTTPS is 2481.
7. Configure these properties:
   - **Synchronization Cache Size** – sets the maximum cache size for the synchronization port. The default is 50MB.
   - **Thread Count** – sets the number of worker threads used for synchronization. The default is 5. If you experience performance issues, you may need to increase this value.
8. (Optional) Expand the optional properties section to configure these properties:

   Note: You can only configure the encryption properties on the primary Unwired Server. Secondary servers will inherit the values, where they become read-only.

   - Certificate Password – is used to decrypt the private certificate listed in certificate file. You specify this password when you create the server certificate.

- Certificate – identifies the location of the security certificate used to encrypt and decrypt data transferred using SSL.
- E2E Encryption Type – specify the asymmetric cipher used for key exchange for end-to-end encryption. You can only use RSA encryption.
- E2E Encryption Certificate – specify the file containing the private key that acts as the identity file for Unwired Server.
- E2E Encryption Certificate Password – set the password to unlock the encryption certificate.
- User Options – sets the command line options for starting the synchronization server. These options are appended the next time the synchronization server starts. These are the available user options:

Option	Description
`@ [variable | filePath]`	Applies listener options from the specified environment variable or text file.
`-a <value>`	Specifies a single library option for a listening library.
`-d <filePath>`	Specifies a listening library.
`-e <deviceName>`	Specifies the device name.
`-f <string>`	Specifies extra information about the device.
`-gi <seconds>`	Specifies the IP tracker polling interval.
`-i <seconds>`	Specifies the polling interval for SMTP connections.
`-l <"keyword=value;...">`	Defines and creates a message handler.
`-m`	Turns on message logging.
`-ni`	Disables IP tracking.
`-ns`	Disables SMS listening.
`-nu`	Disables UDP listening.
`-o <filePath>`	Logs output to a file. **Note:** Ensure that you enter the absolute file path for this property.
`-os <bytes>`	Specifies the maximum size of the log file.
`-ot <filePath>`	Truncates a file, then logs output to that file.

Option	Description
`-p`	Allows the device to shut down automatically when idle.
`-pc [+ | -]`	Enables or disables persistent connections.
`-r <filePath>`	Identifies a remote database involved in the responding action of a message filter.
`-sv <scriptVersion>`	Specifies a script version used for authentication.
`-t [+ | -] <name>`	Registers or unregisters the remote ID for a remote database.
`-u <userName>`	Specifies a synchronization server user name.
`-v [0 | 1 | 2 | 3]`	Specifies the verbosity level for the messaging log.
`-y <newPassword>`	Specifies a new synchronization server password.

Do not use the User Options property in Sybase Control Center to pass in these options: `-c`, `-lsc`, `-q`, `-w`, `-x`, `-zs`.

For more information on synchronization server command line options, see *Listener options for Windows* in the *SQL Anywhere 11.0.1* online help.

9. Click **Save**.

Configuring Messaging-Based Synchronization Properties

Configure one or more synchronization ports to receive service requests from devices.

1. Open Sybase Control Center.
2. In the left navigation pane, expand the **Servers** folder and select a server.
3. Select **Server Configuration**.
4. In the right administration pane, click the **Messaging** tab, and select **Synchronization Listener**.
5. Enter the synchronization port number. The default is 5001.
6. (Optional) Select **Listen on multiple synchronization ports** and enter the additional port numbers.

   Depending on your environment, listening on multiple synchronization ports may provide greater flexibility and reliability. High activity on particular ports, such as virus detection and data inspection, may result in dropped packets or connections if alternate ports are unavailable. When multiple ports are configured, all messaging traffic is still funneled to a single listener.
7. Click **Save**.

Configuring Unwired Server Performance Properties

To optimize Unwired Platform performance, configure the thread stack size, maximum and minimum heap sizes, user options, and inbound and outbound messaging queue counts.

1. Open Sybase Control Center.
2. In the left navigation pane, expand the **Servers** folder and select a server.
3. Select **Server Configuration**.
4. In the right administration pane, select the **General** tab.
5. From the menu bar, select **Performance Configuration**.
6. Configure these properties, as required:

   - Host Name – the name of the machine where Unwired Server is running.
   - Thread Stack Size – the JVM `-Xss` option.
   - Minimum Heap Size – the minimum size of the JVM memory allocation pool, in megabytes. Sybase recommends that this value not fall 500 megabytes for a 32-bit operating system, but 1 gigabyte is recommended. For a 64-bit operating system, Sybase recommends 1 gigabyte for a normal configuration, but 2 gigabyte for a stress configuration (which can vary depending on what RAM is available).
   - Maximum Heap Size – the maximum size of the JVM memory allocation pool, in megabytes. For a 32-bit operating system, Sybase recommends a 1.5 gigabyte maximum heap size value. For a 64-bit operating system, Sybase recommends 1 gigabyte for a normal configuration, but 4 gigabyte for a stress configuration (which can vary depending on what RAM is available).

   Note: The synchronization differencing algorithms are a key feature of RBS; this technology runs in the JVM. You must provide adequate memory to these components. If these algorithms are memory starved, the JVM spends an inordinate amount of time garbage collecting memory, and synchronizations back up in the internal queues. You can monitor process memory usage with tools like SysInternal's Process Explorer to determine the actual amount of memory in use by Unwired Platform, and adjust the JVM heap size accordingly

   Note: Always leave 4 gigabytes for the running of the OS and other applications that may exist on the server.
7. (Optional) Expand the **Show optional properties** section and configure these properties, as required:

   - User Options – other JVM options. For example, you can enable JVM garbage collection logging by setting `-XX:+PrintGCDetails`.
   - Inbound Messaging Queue Count – the number of message queues used for incoming messages from the messaging-based synchronization application to the server. Sybase recommends a choose a value that represents at least 10% of active devices.

- Outbound Messaging Queue Count – the number of message queues used for outbound messages from the server to the messaging-based synchronization application. Sybase recommends a choose a value that represents at least 50% of active devices. However, if you are running 32-bit operating system, do not exceed a value of 100% of active devices.
- Subscribe Bulk Load Thread Pool Size – the maximum number of threads allocated to initial bulk load subscription operations. The default value is five. Setting the thread pool size too high can impact performance.

Note: If you increase either queue count property, ensure you also increase the MaxThread property in the `<hostname>_iiop1.properties` file.

8. Click **Save**.
9. If your server is installed as a Windows service:
   a) Stop Unwired Server.
   b) Open a command prompt.
   c) Run `sup-server-service.bat remove`.
   d) Run `sup-server-service.bat install auto`.
   e) Restart Unwired Server.

See also

- *Configuring the IIOP Socket Listener* on page 143
- *Synchronization Model Performance Tuning* on page 37

Applying Performance Tuning Changes if Unwired Server is a Service

Certain Unwired Server tuning changes require additional steps to apply the changes.

If you installed Unwired Server as a Windows service, then follow these recommendations after changing the configuration of certain properties.

For the CDB pool size, no additional action is required.

1. For Java min/max heap size, and Java thread stack size properties:
   a) Change the property values in Sybase Control Center.
   b) Stop Unwired Server.
   c) Open a command prompt.
   d) Run `sup-server-service.bat remove`.
   e) Run `sup-server-service.bat install auto`.
   f) Restart Unwired Server.
2. For DCN listener thread count, restart the server.

See also

- *Synchronization Model Performance Tuning* on page 37

Messaging Content Sizes

A new property was added to support messaging performance. The message size limitation can be changed by modifying an internal Unwired Server property, `sub.msg.max_content_size`. If you feel the 20KB value should be changed, work with your Sybase representative.

The current message size limit for Unwired Server is 20KB. In general, enlarging the message size results in a lower number of messages, and higher efficiency.

Performance also depends on the device environment. A message that is too large stresses the device, and negates efficiency. Device factors include memory and size of the object graph being sent. In some cases, a larger message size terminates message processing. When the Unwired Server message size exceeds the limit, the message is immediately sent to the client side.

SNMP Notifications

You can set up Sybase Unified Agent to include a Simple Network Message Protocol (SNMP) plug-in that sends notifications to the configured target when the state of an Unwired Server changes (that is, from running to stopped, or from stopped to running).

SNMP is the standard protocol for managing networks and exchanging messages. If the SNMP plug-in is set up for a Sybase Unified Agent, the plug-in creates notifications in response to predetermined status change events that are detected and signaled by the Unwired Server code. When the plug-in generates a notification, a single copy of the notification is transmitted to each target. The SNMP notification target must be the host name and port of a network monitoring station (NMS) that has the ability to process SNMP notifications; Unwired Platform does not include this functionality. Targets and other notification configuration information are read when the Sybase Unified Agent is initialized; therefore, you must stop and restart the agent when enabling SNMP.

Setting Up SNMP Notifications

Setting up SNMP notifications requires you to modify the configuration for Sybase Unified Agent and correctly configure an existing SNMP network monitoring station (NMS). Always test the implementation to validate its setup.

1. *Enabling SNMP Notifications for Unwired Platform*

   Enable the Unwired Platform SNMP plug-in for Sybase Control Center and set up a notification target to receive messages when the status of a local Unwired Server changes.
2. *Handling Transmitted SNMP Notifications*

   Configure an SNMP notification handling tool to process Unwired Platform SNMP notifications.
3. *Testing Notifications with SNMP Queries*

Test the configuration of the Unwired Platform SNMP plug-in for Sybase Control Center by using a management information base (MIB) browser tool to execute an SNMP query.

Enabling SNMP Notifications for Unwired Platform

Enable the Unwired Platform SNMP plug-in for Sybase Control Center and set up a notification target to receive messages when the status of a local Unwired Server changes.

Prerequisites

Before modifying the SNMP `agent-plugin.xml` and `service-config.xml` files, stop the Sybase Unified Agent service.

Task

Note: The SNMP plug-in for Sybase Control Center detects the status of only the Unwired Server running on the same host computer as your instance of Sybase Control Center.

1. Stop Sybase Unified Agent.
2. Enable the SNMP plug-in to run when Sybase Control Center starts:
   a) Open `<UnwiredPlatform_InstallDir>\<SCC-XX>\plugins\com.sybase.supsnmpplugin_1.5.2\agent-plugin.xml`.
   b) Set `register-on-startup="true"`.
   c) (Optional) Modify the value of the `sup.server.ping.schedule.interval` property to specify how often (in seconds) the SNMP plug-in pings Unwired Server to detect the server status. The default is 100.
3. (Optional) Change the SNMP notification target to a custom destination:
   a) Open `<UnwiredPlatform_InstallDir>\<SCC-XX>\services\Snmp\service-config.xml`.
   b) Modify the `snmp.notification.targets` property as follows:

      ```
      set-property property="snmp.notification.targets"
      value=<hostname or IP>/<port number>
      ```

      For example, `set-property property="snmp.notification.targets" value="127.0.0.1/162,10.42.33.136/49152"`. `<hostname or IP>` indicates the server where the network monitoring station (NMS) is located. `<port number>` specifies the SNMP notification port of the NMS. The default SNMP notification port is 162. As indicated in the example, you can set multiple SNMP notification targets.
4. Restart the agent.

Handling Transmitted SNMP Notifications

Configure an SNMP notification handling tool to process Unwired Platform SNMP notifications.

Prerequisites

Install an SNMP notification handling tool, such as HP OpenView.

Task

1. On the SNMP notification target host computer, launch an SNMP notification handling tool.
2. Using the third-party documentation, configure the SNMP notification handling tool to process Unwired Platform SNMP notifications. Configure the notification handler to listen for notifications on the port specified in the "snmp.notification.targets" property of the `<UnwiredPlatformInstall>\<SCC-XX>\services\Snmp\service-config.xml` file.

Testing Notifications with SNMP Queries

Test the configuration of the Unwired Platform SNMP plug-in for Sybase Control Center by using a management information base (MIB) browser tool to execute an SNMP query.

Prerequisites

Install a third-party MIB browser tool.

Task

1. Load `<UnwiredPlatformInstall>\<SCC-XX>\plugins\com.sybase.supsnmpplugin_1.5.2\SYBASE-SUP-MIB.txt` as a module into your MIB browser.
2. Configure the MIB browser settings to use an SNMPv3 information module with the parameters specified in the `<UnwiredPlatformInstall>\<SCC-XX>\services\Snmp\service-config.xml` file:

Property name	Description	Default value
snmp.transport.mappings	SNMP agent host/port	UDP (0.0.0.0/1498) The host "0.0.0.0" represents the local host. Changing this value to a different IP or host name causes the service to fail, since the SNMP service functions only on the local host. The default port number is 1498. You can set a different port for SNMP queries, however, you must ensure that it is not occupied by another SNMP agent.
snmp.usm.user	User name	snmpadmin
snmp.auth.passphrase	Auth password	Sybase4me

3. In the MIB browser, set the **Auth Protocol** to SHA and the **Security Level** to `Auth,NoPriv`.
4. In the object identifier tree of the MIB browser, navigate to `SYBASE-MIB \enterprises\sybase\sup\supObjects\supStatusTable \supStatusEntry` and select **get SNMP variable**.
   Unwired Server status information appears in the data console.

Domain Administration Overview

The goal of domain management is to create and manage domains for one specific tenant. Use multiple domains for multiple tenants sharing the same Unwired Server cluster.

Multiple domains in a cluster allow tenants' administrators (that is, domain administrators) to each manage their own application components. Domain administration for the platform administrator is typically an infrequent administration task that occurs each time a new domain needs to be added to support a change in the tenancy strategy used or need to make changes to an existing domain.

Domains give you the means to logically partitioning environments, thereby providing increased flexibility and granularity of control over domain-specific applications. Administration of multiple customer domains takes place within the same Unwired Platform cluster.

- An Unwired Platform administrator adds and configures domains, creates security configurations for customer applications, and assigns those security configurations to the domain so they can be mapped to packages in the domain. You can use a dedicated security repository like LDAP or the tenant's own provider via a secure VPN-like connection. See *System Administration > Security Administration*.

- One or more domain administrators then perform domain-level actions within their assigned domains.

In a development environment, domains allow developers from different teams to share a single Unwired Server cluster without disrupting application deployment. Administrators can facilitate this by:

1. Creating a domain for each developer or developer group.
2. Granting domain administration privileges to those users so they can perform deployment tasks within their assigned domains.

Table 10. Domain management tasks

Task	Frequency	Administrator
Create domains	Once for each customer	Unwired Platform administrator
Create and assign security configurations, and map roles at package or domain levels	Infrequent, as required	Unwired Platform administrator
Assign and unassign domain administrators	Infrequent, as required	Unwired Platform administrator
Monitor domain logs	Routine	Unwired Platform administrator and domain administrator
Deploy packages	Routine	Unwired Platform administrator and domain administrator
Manage server connections and templates	Infrequent, as required	Unwired Platform administrator and domain administrator
Manage subscriptions	Routine	Unwired Platform administrator and domain administrator
Review client log and MBO/operation error history	Routine	Unwired Platform administrator and domain administrator

See also

- *Multitenant Environments* on page 19

Enabling a Multitenancy Environment with Domains

Platform administrators can add new domains to the Unwired Platform environment to facilitate tenants' administration of their own components.

By default, Unwired Platform uses the Default domain. A single domain does not offer a logical partioning of the mobility environment, which is crucial if you need to support multiple tenants. The number of domains you need to add is determined by the strategy you employ.

Once the setup is complete, domain administrators can manage domain artifacts.

1. *Determining a Tenancy Strategy*

   Determine how many domains to create and how to distribute domain components. A strategic multitenant structure for the cluster balances system-availability considerations with administrative concerns.
2. *Creating and Enabling a New Domain*

   Create and configure multiple domains within a single Unwired Platform installation. A domain must be enabled for application users to access the packages deployed in the domain. Enabling a domain also triggers synchronization of the domain changes to the secondary nodes in the cluster. Application users who attempt to access a disabled domain receive an error message.
3. *Creating a Security Configuration for a Domain*

   Define a set of security providers in Sybase Control Center to protect domain resources, according to the specific security requirements of the domain. A security configuration can either be created first and then mapped to the desired domain.
4. *Activating a Domain Administrator*

   A platform administrator must create and register domain administrators, before this individual can assign the domain administrator to a domain.
5. *Assigning Domain Administrators to a Domain*

   Assign domain administration privileges to a domain administrator. You must be a platform administrator to assign and unassign domain administrators.

Determining a Tenancy Strategy

Determine how many domains to create and how to distribute domain components. A strategic multitenant structure for the cluster balances system-availability considerations with administrative concerns.

Domains are primarly containers for packages for a specific group of users. This group is called a tenant and can be internal to a single organization or external (for example, a hosted mobility environment).

Packages are attached to named security configurations that determine which users have access to mobile business object data, so you must create at least one security configuration

and assign it to the domain for which the package is being deployed. You must identify which users require access to each package and how you will distribute the packages across the system using domains to logically partition the environment.

1. Organize device users according to the data they need to access. Ideally, create a domain for each distinct set of users who access the same applications and authenticate against the same back-end security systems. If you do not need to support multiple groups in distinct partitions, then the single default domain should suffice.
2. Consider how these groups will be affected by administrative operations that prevent them from synchronizing data. Sometimes, you can limit the number of users affected by administration and maintenance disruptions by distributing packages across additional domains. Operationally, the more components a domain contains, the more clients who are unable to access package data during administrative operations like domain synchronizations.
3. Assess the administrative resources of the tenant to determine how much time can be committed to domain administration tasks. Certain multitenant configurations require a greater amount of administrative time. For example, if you distribute packages from the same EIS across a number of domains, you must maintain identical data source configurations for each of these packages. This option requires more administrative time than grouping all packages belonging to the same EIS into one domain.
4. Decide how many domains to create for the customer, and identify which packages to group within each domain, according to the needs of the user groups you identified in step 1.

Creating and Enabling a New Domain

Create and configure multiple domains within a single Unwired Platform installation. A domain must be enabled for application users to access the packages deployed in the domain. Enabling a domain also triggers synchronization of the domain changes to the secondary nodes in the cluster. Application users who attempt to access a disabled domain receive an error message.

Prerequisites

Create a security configuration for the domain and register the domain administrator.

Task

1. Open Sybase Control Center.
2. In the left navigation pane, expand the **Domains** folder.
3. In the right administration pane, select the **General** tab, and click **New**.
4. In the Create Domain dialog, enter a name for the domain and click **Next**.
5. Optional. Select a security configuration for the domain by checking an option from the list of available configurations.

6. Click **Next**.
7. Optional. Select one or more domain administrators for the domain.
8. Click **Finish**.
   The new domain appears in the **General** tab.
9. Click the box adjacent to the domain name, click **Enable**, then click **Yes** to confirm.

Creating a Security Configuration for a Domain

Define a set of security providers in Sybase Control Center to protect domain resources, according to the specific security requirements of the domain. A security configuration can either be created first and then mapped to the desired domain.

A security configuration determines the scope of data security. A user must be part of the security repository used by the configured security providers to access any resources (that is, either a Sybase Control Center administration feature or a data set from a back-end data source) on a domain. See *Sybase Control Center online help > Configure > Configuring Unwired Platform > Security Configurations*.

1. In Sybase Control Center, add a new security configuration using the **Security** node.
2. In the left navigation pane, expand the **Security** folder and select the new configuration.
3. Use the **Authentication**, **Authorization**, **Attribution**, and **Audit** tabs to configure the appropriate security providers for each aspect of domain security.
4. Edit the security provider properties, as required.
5. Validate the configuration to ensure that Unwired Server accepts the changes.
6. Apply the changes to Unwired Server.

Activating a Domain Administrator

A platform administrator must create and register domain administrators, before this individual can assign the domain administrator to a domain.

Prerequisites

Domain administrator physical roles should already be mapped to the default SUP Domain Administrator logical role per mapping in 'admin' security configuration in 'default' domain. However, this role name may differ if you choose to use a domain administration role from your company's repository.

Task

1. In the Security node of Sybase Control Center, create a new administrator user by providing the: login, company name, first name, and last name.

   See *Sybase Control Center online help > Configure > Configuring Unwired Platform > Domains > Domain Administration > Registering a Domain Administrator User*.
2. Assign the login the required physical role in the security provider repository that authenticates and authorizes administrative logins for Sybase Control Center and Unwired

Server. See *Sybase Control Center online help > Configure > Configure Sybase Control Center > Authorization*.

Assigning Domain Administrators to a Domain

Assign domain administration privileges to a domain administrator. You must be a platform administrator to assign and unassign domain administrators.

Prerequisites

Ensure the user is already registered as a domain administrator in the Domain Administrators tab.

Task

1. Open Sybase Control Center.
2. In the left navigation pane, expand the **Domains** folder, and select the domain for which to assign domain administration privileges.
3. Select the domain-level **Security** folder.
4. In the right administration pane, select the **Domain Administrators** tab, and click **Assign**.
5. Select one or more administrator users to assign to the domain by checking the box adjacent to the user name.
6. Click **OK**.
   A message appears above the right administration pane menu indicating the success or failure of the assignment. If successful, the new domain administrator appears in the list of users.

Managing and Maintaining Domains

Configure domain components, including datasource connections, logging, role mappings, and packages.

These tasks can be performed by both platform and domain administrators.

1. *Creating Data Source Connections*

   A connection is required to send queries to mobile business objects and receive data. Configure the properties required to connect to EIS datasources.
2. *Enabling and Configuring Domain Logging*

   Activate or deactivate domain logging in Sybase Control Center, and configure domain log autopurge settings for all nodes in a cluster. Domain logging collects data that pertains to the activities of all packages in a domain. You must have administrator privileges to configure domain logging.
3. *Mapping Roles for a Domain*

Configure role mapping in Sybase Control Center to manage logical roles and authorize client requests to access domain resources.

4. *Deploying an MBO Package to a Domain*

   Deploy an MBO package to Unwired Server using Sybase Control Center. Packages are initially created by developers, but are deployed and maintained on a production Unwired Server by administrators.

5. *Deploying a Mobile Workflow Package to a Domain*

   Use Sybase Control Center to deploy a mobile workflow package to make it available on Unwired Server. Packages are initially created by developers, but are deployed and maintained on a production Unwired Server by administrators.

Creating Data Source Connections

A connection is required to send queries to mobile business objects and receive data. Configure the properties required to connect to EIS datasources.

The format in which data is communicated depends on the type of datasource. Establish connections by supplying an underlying driver and a connection string that allow you to address the datasource, and provide you a mechanism by which to set the appropriate user authentication credentials and connection properties. See *Sybase Control Center online help > Configure > Configuring Unwired Platform > Connections*.

Note: When creating connections, please ensure that the prerequisites for each connection type are installed on all nodes of the cluster.

1. Open Sybase Control Center.
2. Select the domain-level **Connections** node for the domain to configure.
3. Create a new connection or connection template by selecting the appropriate tab and clicking **New**.
4. Enter a unique connection pool name, and select both a connection pool type and the appropriate template to use for that type. Customize the template, if required, by editing existing values or adding new properties.
5. Test the values you have configured by clicking **Test Connection**. If the test fails, either the values you have configured are incorrect, or the datasource target is unavailable. Evaluate both possibilities and try again.
6. Click **OK** to register the connection pool.
   The name appears in the available connection pools table on the Connections tab; administrators can now use the connection pool to deploy packages.

Enabling and Configuring Domain Logging

Activate or deactivate domain logging in Sybase Control Center, and configure domain log autopurge settings for all nodes in a cluster. Domain logging collects data that pertains to the

activities of all packages in a domain. You must have administrator privileges to configure domain logging.

First, domain-level logging must be enabled by a platform administrator. Domain-level logging controls whether package-level logging captures data. Then either the platform administrator or the domain administrator can enable logging on a per-package basis from the Packages node of Sybase Control Center. See *Sybase Control Center online help > Configure > Configuring Unwired Platform > Packages > Enabling Package Logging*.

1. Open Sybase Control Center.
2. In the left navigation pane, expand the **Domains** folder and select the domain for which to configure log settings.
3. Select **Log**.
4. In the right administration pane, select the **Settings** tab.
5. Select one of:
   - **Enable** – activate domain logging in Sybase Control Center.
   - **Disable** – turn off domain logging.
6. Set the autopurge threshold by entering the length of time (in days) to retain domain log data.
7. Click **Save**.

Mapping Roles for a Domain

Configure role mapping in Sybase Control Center to manage logical roles and authorize client requests to access domain resources.

Prerequisites

Unwired Platform cannot query all enterprise security servers directly; to perform authentication successfully know the physical roles that are required.

Task

Typically, the endpoint and role mapping used by developers are not the same as for a production system. Administrators must reset these configurations accordingly. When you map domain-level roles, these roles are automatically applied to the packages that use the same security configuration.

1. Expand the domain-level Security node and select the security configuration to map roles for.
2. Set an appropriate mapping state for each logical role:
   - **NONE** – disable logical roles.
   - **AUTO** – allow logical roles to be dynamically mapped.

- **Map Roles** – manually map required physical roles for a logical role when physical and logical role names do not match. If names do not match, the AUTO mapping state does not work; mappings cannot occur dynamically.

The states of AUTO or NONE require the least administration.

3. To manually map roles, use the Role Mappings dialog to map a logical role to one or more physical roles. You can also map multiple logical roles to the same physical role. Add or delete available roles as needed.
   Once a logical role is manually mapped, the mapping state changes to MAPPED. Mapped roles appear in the active Physical Roles cell in the domain-wide role mappings table.

See also

- *Roles and Mappings* on page 112

Deploying an MBO Package to a Domain

Deploy an MBO package to Unwired Server using Sybase Control Center. Packages are initially created by developers, but are deployed and maintained on a production Unwired Server by administrators.
In the left navigation pane, from the **Domain** node, launch the Deploy wizard. See *Deploying a Replication or Messaging Package* in Sybase Control Center online help.

Deploying a Mobile Workflow Package to a Domain

Use Sybase Control Center to deploy a mobile workflow package to make it available on Unwired Server. Packages are initially created by developers, but are deployed and maintained on a production Unwired Server by administrators.
In the left navigation pane, from the **Workflows** node, launch the Deploy wizard for mobile workflow packages. See *Deploying a Mobile Workflow Package* in Sybase Control Center online help.

EIS Connection Management Overview

The goal of enterprise information system connection management is to ensure the connections to back-end repositories of data remain available to Unwired Server and deployed packages that require those connections. Connections management is a non-routine administration task.

Review the tasks outlined in this table to understand the data management workflow for each role and the degree of activity it entails.

Task	Frequency	Accomplished by
Create and tune connections	On-demand as needed	Sybase Control Center for Unwired Platform with the Connections node

Task	Frequency	Accomplished by
Create and maintain connection pool templates	One-time	Sybase Control Center for Unwired Platform with the Connections node
Update package connections when moving from development and test environments to production environments	On-demand, as needed	Sybase Control Center for Unwired Platform with the **Deployment Wizard**

See also

- *EIS Connections* on page 30
- *EIS Data Source Connection Properties Reference* on page 308

Data Source Connections

A data source connection is a physical connection definition that provides runtime connection to enterprise information systems (EIS), that in turn enables data to be mobilized by Unwired Server to client device via synchronization or messaging. Before you create publications or subscriptions, or deploy packages, you must first define database connections.

For Unwired Server to recognize a EIS data source, you must define a connection to that data repository. The connections are defined in Sybase Control Center with the Unwired Platform perspective, and are known as server-to-server connections because they are opened by Unwired Server.

In Unwired Platform you create a connection template from which you can replicate connections. Connection pools allows Unwired Servers to share pools of pre-allocate connections to a remote EIS server. This preallocation avoids the overhead imposed when each instance of a component creates a separate connection. Connection pooling is only supported for database connections, and the size of the pool is controlled by the Max Pool Size property of the connection.

Note: When creating connections, ensure that the prerequisites for each connection type are installed on all nodes of the cluster. See System Administration guide > Environment Setup > EIS Connections.

See also

- *Data Source Connection Reference* on page 308

Connection Templates

A connection template is a model or pattern used to standardize connection properties and values for a specific connection pool type so that they can be reused. A template allows you to quickly create actual connections.

Often, setting up a connection for various enterprise data sources requires each administrator to be aware of the mandatory property names and values for connecting to data sources. Once you create a template and add appropriate property names and corresponding values (for example user, password, database name, server name, and so on), you can use the template to instantiate actual connection pools with predefined property name and value pairs.

CHAPTER 8 Device Provisioning

Provisioning describes how to setup the device, so it can interact with the Unwired Platform environment. Device provisioning assumes that the user subscriber accounts have already been setup in Sybase Control Center.

Device provisioning is supported differently, depending on various mobility environment factors:

Method	Deployment	Device types	Unwired Platform components
Cradles with desktop device managers	Personal deployment [1]	Windows, Windows Mobile, and BlackBerry	For runtime clients (messaging/replication) and device applications.
Afaria Over-the-air (OTA)	Enterprise deployment in production environments	Windows, Windows Mobile, BlackBerry	For runtime clients (messaging/replication) and device applications. This type of provisioning is typically jointly performed by the administrator (back-end setup to OTA environment) and the device user (installation on the device) after the notification is received.
iTunes	Enterprise deployment with an Apple enterprise certificate	iPhone	For complete packaged iPhone applications that are distributed by businesses to internal users — especially when users do not have an App Store account.

Device provisioning includes:

1. Distributing and installing the platform infrastructure components. Devices that require provisioning of infrastructure components include:
   - Afaria clients for Windows, Windows Mobile, and BlackBerry devices.
   - Client runtime for messaging-based synchronization applications for Windows and Windows Mobile. No runtime is required for iOS.
2. Distributing and installing the device application that interacts with MBO packages that are deployed to the Unwired Server. For supported device types, administrators can use

[1] Either development/test or small-scale production environments

Afaria. For unsupported types (like iOS), administrators must review the device's user guide to see how application provisioning might occur.

3. If you are using a replication base synchronisation application, setting up the device for push synchronization.
4. If you are using a messaging-based synchronization package, registering the user devices using Sybase Control Center (each user then activates using the Sybase Settings application deployed in step 1).

Runtimes and Clients

In Unwired Platform there is a difference between runtime and clients.

There are two parts to setting up a device as a client to Unwired Platform, depending on the device type and the application you require for your mobile environment:

Client software	Description
Unwired client runtimes and applications	Administrators must provision the Unwired client runtime files and application binaries along with the Afaria client if necessary. The device-platform client runtime files for native and workflow applications are located in: `<UnwiredPlatform_InstallDir>\ClientAPI\`.
Afaria clients	Afaria clients are included with Afaria. Administrators must download and install one of these clients on the device if you want to provision devices with Afaria over-the-air (OTA); until then, administrators and developers can provision individual devices sequentially by cradling a mobile device and using the device's desktop manager.

See also

- *Configuring Push Notifications for the BlackBerry Enterprise Server* on page 175
- *Afaria Provisioning and Mobile Device Management* on page 166
- *Apple Provisioning for iOS* on page 171
- *BES and BIS Provisioning for BlackBerry* on page 174
- *Setting up Push Synchronization for Replication Synchronization Devices* on page 177
- *Create and Deploy Afaria Clients and Unwired Platform Runtimes* on page 170

Afaria Provisioning and Mobile Device Management

Afaria extends Unwired Platform functionality by providing additional device client management features for remote and mobile computing devices, including laptops, desktops, and handheld devices.

Use the Afaria tools to:

- Deploy client and runtime infrastructure components over-the-air
- Track assets
- Secure devices and data

See also

- *Runtimes and Clients* on page 166
- *Apple Provisioning for iOS* on page 171
- *BES and BIS Provisioning for BlackBerry* on page 174
- *Setting up Push Synchronization for Replication Synchronization Devices* on page 177

Launching Afaria from Sybase Control Center

If you purchased Afaria, you can open the Afaria® Web interface administration console to manage Afaria resources directly from Sybase Control Center. This optional step makes Afaria a managed resource of Sybase Control Center.

Prerequisites

Ensure the Afaria Server is running by checking the Windows Services. You cannot open the Afaria Administration console unless this server is running.

Task

1. From the Sybase Control Center menu, click **Resource** > **Register**.
2. Configure the **Resource Type** and **Connection Information** properties.
   For example, if Sybase Control Center is installed on the same computer as Afaria Server, use the following connection properties:

Options	Description
Host	localhost
Port	80

   The Afaria server is added to the **Perspective Resources** window, and takes the display name you configured for it.
3. In the **Perspective Resources** window, right-click the Afaria Server you want to display the administration console for and select **Manage**.

If you do not perform this step, you can still lauch Afaria from its desktop shortcuts. See Afaria documentation for details.

Setting Up the Afaria Environment

Setting up the Afaria environment in Unwired Platform primarily involves configuring over-the-air (OTA) deployment. OTA deployment requires specific setup of the Afaria server and uses multiple tools to accomplish this task.

Prerequisites

Ensure you have configured Sybase Control Center to open Afaria Administrator. No desktop shortcuts are created with an Unwired Platform installation of Afaria.

Task

Be aware of the following Unwired Platform requirements:

- Only use IIS on Windows as the OTA Deployment Center host when using Afaria with Unwired Platform.
- Install the OTA Deployment Center behind the DMZ and use a Sybase relay server to relay download requests.
- Do not install the deployment center on the same host as either Afaria Administrator or Afaria Server; doing so creates a greater risk of a port conflict, depending on protocols used (by default, the Afaria components share port 80).
- Review the system requirements information Afaria provides.

1. *Setting Up the OTA Deployment Center and the SMS Gateway*

   The OTA Deployment Center allows device users to access and download Afaria clients and runtimes over-the-air, rather than needing cradle the device and connect to the network. OTA deployment is recommended for production environments that require you to administer many device users.
2. *Configuring Afaria Server*

   After you have set up OTA Deployment Center and configured either an SMS or SMTP gateway, configure Afaria Server.
3. *Creating Addresses, Groups, and Profiles*

   Session Manager allows you to control each user session opened with the SMS or SMTP download notification and determine what actions are subsequently performed.
4. *Create and Deploy Afaria Clients and Unwired Platform Runtimes*

   When the user receives a client notification, they download the installer and install the Afaria client and any required Unwired Platform runtime files.

Setting Up the OTA Deployment Center and the SMS Gateway

The OTA Deployment Center allows device users to access and download Afaria clients and runtimes over-the-air, rather than needing cradle the device and connect to the network. OTA

deployment is recommended for production environments that require you to administer many device users.

1. Install the Afaria SMS gateway, which is a required communication path for the OTA Deployment Center.

   See *Afaria Reference Manual | Platform > Server Configuration > Properties > SMS Gateway Installation*.
2. Set up the OTA Deployment Center.

   See *Installing Afaria > Setting up the OTA Deployment Center*. For details about requirements, see *Afaria Release Notes > Component and Feature Requirements*.

Configuring Afaria Server

After you have set up OTA Deployment Center and configured either an SMS or SMTP gateway, configure Afaria Server.

1. Open Afaria Administrator and configure Afaria Server to use the OTA Deployment Center and SMS gateway:
   - **Server Configuration > Properties > OTA Deployment Center** to configure settings for this component. If you are using a relay server, ensure that you configure values used by this Unwired Platform component correctly.

     See *Afaria Reference Manual | Platform > Server Configuration > Properties > OTA Deployment*.
   - **Server Configuration > Properties > SMS Gateway** to enable the SMS channel, which is a required communication path for the OTA Deployment Center.

     See *Afaria Reference Manual | Platform > Server Configuration > Properties > SMS Gateway* and *Afaria Reference Manual | Platform > Server Configuration > Properties > Addresses and routing for Afaria messages*.
2. (Optional) Configure an SMTP gateway to send SMS messages over an e-mail infrastructure, which may be used for e-mail-enabled messaging devices.

   See *Afaria Reference Manual | Platform > Server Configuration > Properties > SMTP*.
3. Restart Afaria Server to implement these server configuration changes.

Creating Addresses, Groups, and Profiles

Session Manager allows you to control each user session opened with the SMS or SMTP download notification and determine what actions are subsequently performed.

The session manager channel is used only after the Afaria client files have been correctly installed on the device.

For complete details and references to required related topics, see *Afaria Reference Manual | Platform > Administration > Profile*, *Afaria Reference Manual | Components > Session Manager and Software Manager* chapters.

1. In Afaria Administrator, create address book entries to define contact information for devices so that users can be contacted, using the SMS or the SMTP gateway you have enabled.

   Note: If you have a user list in another application and that application allows you to export entries to a `*.CSV` file, you can import this information into Afaria Administrator. See *Afaria Reference Manual | Platform > Home > Client Deployment > Address Book Properties, Distribution List Properties, and Importing Addresses*.

2. Create groups, and assign users to them.
3. Create profiles that define what happens during the user session (for example, what instructions are given, which items are sent to a client, and so on).
4. Assign these groups (and therefore its members) to the session channel.

Create and Deploy Afaria Clients and Unwired Platform Runtimes

When the user receives a client notification, they download the installer and install the Afaria client and any required Unwired Platform runtime files.

1. Use the Afaria Create Client Installation wizard to create a client installer. This program is located only on the Afaria server: `<AfariaServerInstallDir>\Bin\XSClientInstall.exe`.

   A new Afaria client relies on connection configuration settings to connect back to the Afaria Server and run its first session. The Create Client Installation wizard creates seed data and stores it on the client as connection settings.

   CAB files must be signed and placed in the installation sequenced required. Otherwise, the user receives SMS or SMTP messages that link to the OTA Deployment Center in an incorrect order. See *Afaria Reference Manual | Platform > Creating Clients > Creating Afaria Clients*, and refer to the program's context-sensitive help as required.

   Note: If you select the **Companion PC** option, device users can also install by cradling the device and synchronizing the files and seed data from the corresponding desktop device manager (for example, ActiveSync).

2. To make downloads available, publish components to the OTA Deployment Center. Use program called OTA Publisher, which is installed on the Afaria Server host machine in this location: `<AfariaServerInstallDir>\Bin\OTAPublisher.exe`. For details about how to use the OTA Publisher, see the program's context-sensitive help.
3. Send a device notification to allow the user to trigger the provisioning process.

   See *Afaria Reference Manual | Platform > Home > Client Deployment > Sending Notifications*. Depending on the gateway you configured, that user address is checked and a message is sent using either SMS or SMTP.
4. Validate performance by checking logs.

   See *Afaria Reference Manual | Platform > Data Views > Working with Logged Actions*

See also

- *Runtimes and Clients* on page 166

Apple Provisioning for iOS

Unlike the Afaria method of device provisioning, Apple provisioning is primarily performed by developers at the end of their development cycle, with some Sybase Control Center configuration performed by the administrator, and some device setup by the device user. Apple Push Notification Service (APNS) supports user notifications on iPhone devices.

iOS devices do not require a runtime or a client; only the application must be deployed to the device. The lack of a runtime allows users to self-manage their devices: device users download application files to their iOS devices and synchronize updates as required.

Apple Push Notification Service (APNS) allows users to receive notifications on iPhones. APNS:

- Works only with iPhone physical devices
- Is not required for any iOS application
- Cannot be used on a on an iPhone simulator
- Cannot be used with iPod touch or iPad devices
- Must be set up and configured by an administrator on the server
- Must be enabled by the user on the device

See also

- *Runtimes and Clients* on page 166
- *Afaria Provisioning and Mobile Device Management* on page 166
- *BES and BIS Provisioning for BlackBerry* on page 174
- *Setting up Push Synchronization for Replication Synchronization Devices* on page 177

Configuring Apple Push Notification Service

Use Apple Push Notification Service (APNS) to push notifications from Unwired Server to the iOS application. Notifications can include badges, sounds, or custom text alerts. Device users can customize which notifications to receive through Settings, or turn them off.

Prerequisites

The following prerequisites must be performed by the developer before the administrator can configure the Apple Push Notification Service (APNS):

- Register for the iPhone Developer Program as an enterprise developer to access the Developer Connection portal and get the certificate required to sign applications.

- Create an App ID and ensure that it's configured to use Apple Push Notification Service (APNS).
- Create and download an enterprise APNS certificate that uses Keychain Access in the Mac OS. The information in the certificate request must use a different common name than the development certificate they might already have. This is because the enterprise certificate also creates a private key, which must be distinct from the development key. This certificate must also be imported as a login keychain and not a system key chain and the developer should validate that the certificate is associated with the key in the Keychain Access application. *Get a copy of this certificate.*

  Note: A new 2048-bit Entrust certificate needed for Apple Push Notification Service (APNS) push to work, because APNS push functionality stops working on 22 December 2010.

  Apple now uses a 2048-bit root certificate from Entrust, which provides a more secure connection between Unwired Server and APNS. This certificate comes with the Windows OS, and is upgraded automatically with Windows Update, if it is enabled. This information is not part of the procedure that documents APNS support.

  If Windows Update is disabled, you must manually download and install the certificate. Go to: *https://www.entrust.net/downloads/binary/entrust_2048_ca.cer*. For help on installing the certificate, see *http://www.entrust.net/knowledge-base/technote.cfm?tn=8282*.
- Create an enterprise provisioning profile and include the required device IDs with the enterprise certificate. The provisioning profile authorizes devices to use applications you have signed.
- Create the Xcode project ensuring the bundle identifier corresponds to the bundle identifier in the specified App ID. *Ensure you are informed of the "Product Name" used in this project.*
- Used APNS initialization code in the codeline.

Developers can review complete details in the *iPhone OS Enterprise Deployment Guide* at *http://manuals.info.apple.com/en_US/Enterprise_Deployment_Guide.pdf*.

Task

Each application that supports Apple Push Notifications must be listed in Sybase Control Center with its certificate and application name. You must perform this task for each application.

1. Confirm that the IT department has opened ports 2195 and 2196, by executing:

   ```
   telnet gateway.push.apple.com 2195
   telnet feedback.push.apple.com 2196
   ```

   If the ports are open, you can connect to the Apple push gateway and receive feedback from it.

2. Copy the enterprise certificate (`*.p12`) to the computer on which Sybase Control Center has been installed. Save the certificate in *`UnwiredPlatform_InstallDir`*`\Servers\MessagingServer\bin\`.
3. In Sybase Control Center, expand the **Servers** folder and click **Server Configuration** for the primary server in the cluster.
4. In the **Messaging** tab, select **Apple Push Configuration**, and:
   a) Configure Application name with the same name used to configure the product name in Xcode. If the certificate does not automatically appear, browse to the directory.
   b) Change the push gateway information to match that used in the production environment.
   c) Restart Unwired Server.
5. Verify that the server environment is set up correctly:
   a) Open *`UnwiredPlatform_InstallDir`*`\Servers\UnwiredServer\logs\APNSProvider`.
   b) Open the log file that should now appear in this directory. The log file indicates whether the connection to the push gateway is successful or not.
6. Deploy the application and the enterprise distribution provisioning profile to your users' computers.
7. Verify that the APNS-enabled iOS device is set up correctly:
   a) Click **Device Users**.
   b) Review the Device ID column. The application name should appear correctly at the end of the hexadecimal string.
   c) Select the Device ID and click **Properties**.
   d) Check that the APNS device token has been passed correctly from the application by verifying that a value is in the row. A device token appears only after the application runs.
8. Test the environment by initiating an action that results in a new message being sent to the client.

   If you have verified that both device and server can establish a connection to the APNS gateway, the device receives notifications and messages from the Unwired Server. Allow a few minutes for the delivery or notification mechanism to take effect and monitor the pending items in the Device Users data to make sure the value increases appropriately for the applications.
9. To troubleshoot APNS, use the *`UnwiredPlatform_InstallDir`*`\Servers\Unwired Server\log\trace\APNSProvider` log file. You can increase the trace output by editing `<SUP_Home>\Servers\MessagingServer\Data\TraceConfig.xml` and configuring the tracing level for the APNSProvider module to debug for short periods.

BES and BIS Provisioning for BlackBerry

BlackBerry devices can be provisioned with BlackBerry servers in addition to the Afaria method.

BlackBerry devices that are connected to a production environment using relay server can use either BlackBerry Enterprise Server (BES) or BlackBerry Internet Server (BIS) to provision supported device types.

See also

- *Runtimes and Clients* on page 166
- *Afaria Provisioning and Mobile Device Management* on page 166
- *Apple Provisioning for iOS* on page 171
- *Setting up Push Synchronization for Replication Synchronization Devices* on page 177

Provisioning Prerequisites for BlackBerry

Complete the prerequisites before provisioning the application.

First, look at the prerequisites for all device platforms, then this information for BlackBerry-specific configuration details.

Sybase Unwired Platform Installation and Configuration

Prerequisite	Where to Find Information
Install a Sybase Relay Server if Unwired Server is behind the internal firewall. **Note:** The use of a Relay Server is optional; however, if you want to use one in your Unwired Platform system configuration, you would install and configure it prior to provisioning the application to the device.	*Relay Server Setup*

BlackBerry Enterprise Server Configuration

Prerequisite	Where to Find Information
Configure push notifications from Unwired Server to each BES.	*Configuring BlackBerry Push Settings*
Add users to each BES.	*BlackBerry Enterprise Server Administration Guide*

Prerequisite	Where to Find Information
Generate an activation password, then send account information (e-mail address and password) to device users.	*BlackBerry Enterprise Server Administration Guide*

Preinstallation Device Configuration

Have device users complete the device preinstallation tasks before provisioning the application to the device.

Prerequisite	Where to Find Information
Install device prerequisites.	*Runtimes and Clients* as well as the Device User Guide for any SAP solution you are deploying. For example, *Sybase Mobile Sales for SAP CRM Device Users Guide for BlackBerry > Installation Prerequisites*
Provide enterprise activation information so device users can pair their devices to the BlackBerry Enterprise Server.	*http://na.blackberry.com/eng/support/enterpriseactivation/*
Install BlackBerry Desktop Software on a personal computer, which includes Desktop Manager, if your organization is installing the application on a small number of devices. **Note:** Desktop Manager is required only if your device users will use it to install the application.	*http://na.blackberry.com/eng/services/desktop/*

See also

- *Chapter 5, Relay Server Clusters* on page 53
- *Runtimes and Clients* on page 166
- *Configuring Push Notifications for the BlackBerry Enterprise Server* on page 175

Configuring Push Notifications for the BlackBerry Enterprise Server

Configure push notifications from Unwired Server to the BES using Sybase Control Center.

BlackBerry push notifications use SMS-based push to alert offline users to the availability of new items awaiting retrieval on Unwired Server. SMS-based push uses an IP connection only long enough for the Send/Receive data exchange to complete. The feature overcomes network issues with always-on connectivity and battery life consumption on 3G networks. You can also enable the Push Access Protocol (PAP) if you are using a Push Proxy Gateway to deliver messages.

You configure the notifications for each BES.

1. Log into Sybase Control Center.
2. Expand **Servers** and select the appropriate server.
3. Select **Server Configuration**.
4. In the right administration pane, select the Messaging tab, then select the **BlackBerry Push Notification** tab.
5. Select **New**, then enter all applicable information.

   The URL takes the format:
   `http://<DNS or IP address>:<port number>`

   The default port number is 8080.

See also

- *Chapter 5, Relay Server Clusters* on page 53
- *Runtimes and Clients* on page 166

Provisioning Options for BlackBerry Devices

To provision the application to BlackBerry devices, you can automatically push the application to the device or send a link to device users so they can install it when desired. For small deployments or evaluation purposes, device users can install the application using BlackBerry Desktop Manager.

Once installed on the device, the application appears in Downloads. However, device users can move it to a different location. If device users reinstall the application from a link or URL, or using Desktop Manager, the BlackBerry device remembers the installation location.

Provisioning Method	Purpose	Description
BlackBerry Enterprise Server (BES) Over-the-Air (OTA)	Enterprise installations	When the BlackBerry device activates, it automatically pairs with the BES and downloads the application. See *http://www.blackberry.com/btsc/search.do?cmd=displayKC&docType=kc&externalId=KB03748* for step-by-step instructions.
OTA: URL/link to installation files	Enterprise installations	The administrator stages the OTA files in a Web-accessible location and notifies BlackBerry device users via an e-mail message with a link to the JAD file.
Desktop Manager	Personal installation	Installs the application when the BlackBerry device is synced via a computer.

Provisioning Method	Purpose	Description
BlackBerry App World	Evaluation/Demo	Downloads a solution-specific application like Sybase Mobile Sales for SAP CRM in demo mode. Demo mode runs standalone, and the device does not connect to Sybase Unwired Platform or SAP CRM. *http://appworld.blackberry.com/webstore/*

Setting up Push Synchronization for Replication Synchronization Devices

If your device application requires push synchronization, you must configure the device for server-initiated synchronization.

Prerequisites

The device must already be provisioned with Unwired Platform runtimes. You also must have also enabled push synchronization for the Unwired Server as well as configured cache refresh triggers for the mobile business object in Sybase Control Center. For details, see Sybase Control Center online help:

- *Configure > Configuring the Unwired Platform Environment > Unwired Server > Server Properties > Replication > Enabling Push Synchronization*
- *Configure > Configuring the Unwired Platform Environment > Packages > Setting Up a Cache group > Scheduling a Cache Refresh*

Task

1. The developer creates the application and designs it as a push synchronization application.
2. The user starts the application on the device or emulator.
3. The user or administrator enables push sync on the device or emulator and set the push notification role.
4. The user synchronizes the MBO to Unwired Server, and restarts the listener so future synchronizations push to the device correctly.

See also

- *Runtimes and Clients* on page 166
- *Afaria Provisioning and Mobile Device Management* on page 166
- *Apple Provisioning for iOS* on page 171

- *BES and BIS Provisioning for BlackBerry* on page 174

CHAPTER 9 Package Administration

Unwired Platform packages are administered differently depending on their type. However, all packages types must be deployed and configured.

- Replication-based synchronization mobile business object (MBO) packages use an application layer service that synchronously pulls data changes from the server. Because data updates occur in bursts because of periodic synchronization between the client and server, replication-based synchronization packages use the following package configuration features:
  - Subscription templates are created by the administrator and define how the device user is to be notified when cache data changes, depending on the subscription properties configured for that synchronization group in the package.
  - Cache groups define policies that determines the frequency and the level to which server data is refreshed and used by the packaged MBO. Create as many cache groups as required to meet the varying data refresh needs of the MBOs for a given mobile application.
  - Synchronization groups determine the frequency with which notifications are generated from Unwired Server to initiate MBO synchronization, and thereby access refreshed data. A synchronization group is a collection of MBOs that are synchronized together at regular intervals. Notifications are then delivered, depending on cache group schedule repeat, synchronization group change detection interval, and subscription notification threshold values.
- Message-based synchronization MBO packages create a device service that uses a dedicated channel that is always open to asynchronously push data changes, requests, and notifications between client and server. Because messaging-based MBO packages make use of both replicated and nonreplication (message) data to complete a mobile workflow process, they use these package configuration features:
  - Subscriptions are user-defined and specify the synchronization messages mobile device users receive and how they receive these messages. However, administrators can configure some properties to further refine and control the behavior.
  - Synchronization groups for messaging-based synchronization packages directly download the unit of changed data to the device, as opposed to sending a notification message. A synchronization group is a collection of MBOs that are synchronized together at regular intervals.
- Mobile workflow packages must be created with the Mobile Workflow Application Designer. This tool allows developer to design mobile workflow screens that can call on the create, update, and delete operations, as well as named queries, of a mobile business object.

- E-mail settings allow the administrator to configure a listener to scan all incoming e-mail messages delivered to the particular inbox that the administrator indicates during configuration.
- Matching rules are used by the e-mail listener to identify e-mail messages that match the rules specified by the administrator. When the e-mail messages match the rule, Unwired Server sends the e-mail message as a mobile workflow to the device that matches the rule.
- Context variables customize how data is loaded into the Unwired Server cache. By determining the context variables you want to use, you can create a data set that is smaller, more focused, and therefore yields better performance.

Deployment

The last step of mobile business object (MBO) development is to deploy the MBO definitions to Unwired Server as a deployment unit generated from a design-time deployment package using Sybase Unwired WorkSpace.

When you deploy MBOs to the Unwired Server, you are deploying:

- MBO definitions including attributes, operations, connections, role mappings, schedule groups, cache groups as defined in the package.
- MBO custom code related to object filters, result-set filters, and result checkers.
- Appropriate generated server-side artifacts that support the interaction with the EIS back-ends and device application.
- Other functionality captured in the MBO model.

MBOs are deployed using a deployment wizard through which you can make the choices that are appropriate for application requirements. Developers use Unwired WorkSpace to deploy a package.

The production administrator can deploy from a wizard using the web-based management console, or from the command line. Deployment-time tasks include choosing:

- Target domain – logical container for packages.
- Security configuration – used for authentication and authorization of users accessing the package.
- Role-mappings – to map logical roles to the physical roles of the back-end repository.
- Server connections mapping – to bind MBOs design-time data sources to production data sources.

MBO Package Management Overview

The goal of mobile business object (MBO) package management is to make MBOs available to device users. MBO package management typically requires a one-time deployment and

configuration, except for ongoing subscription management for messaging and Data Orchestration Engine connector (DOE-C) packages.

A package, along with its current settings for cache groups, role mappings, synchronization groups, connections, and security configuration, can be exported to an archive and imported back into Sybase Control Center for backup or to facilitate a transition from a test environment to a production environment.

Table 11. MBO package management tasks

Task	Package type	Frequency	Accomplish by using
Deploy packages to a development or production Unwired Server	RBS and MBS	Once, unless a new version becomes available	Sybase Control Center for Unwired Platform with the domain-level Packages node
Control user access by assigning security configurations for each package, and mapping roles if fine-grained authorization is enforced through logical roles	RBS and MBS	Once, unless security requirements of the package change	Sybase Control Center for Unwired Platform with the domain-level Packages node
Set up the package cache interval and cache refresh schedule (for getting data updated on the Unwired Server from the data source)	RBS and MBS	Once, unless data refreshes need to be tuned	Sybase Control Center for Unwired Platform with the domain-level Packages node
Manage subscriptions (RBS, MBS, and DOE-C), synchronization groups (RBS and MBS), and device notifications (RBS) to customize how updated data in the cache is delivered to the device user	Varies	Periodic, as required	Sybase Control Center for Unwired Platform with the domain-level Packages node
Export or import an MBO package	RBS and MBS	On-demand, as required	Sybase Control Center for Unwired Platform with the domain-level Packages node
Review current/historical/performance metrics	All	Routine	Sybase Control Center for Unwired Platform with the Monitor node (available only to administrators)

Deploying and Managing MBO Packages

Use Sybase Control Center to deploy MBO packages created by developers to a production Unwired Server and manage package configuration.

Multiple tasks are involved in package deployment and management. Package administration tasks vary depending on the type of package you deploy.

Deploying an MBO Package to a Domain

Deploy an MBO package to Unwired Server using Sybase Control Center. Packages are initially created by developers, but are deployed and maintained on a production Unwired Server by administrators.

In the left navigation pane, from the **Domain** node, launch the Deploy wizard. See *Deploying a Replication or Messaging Package* in Sybase Control Center online help.

See also

- *Selecting a Security Configuration for a Package* on page 182
- *Mapping Roles for a Package* on page 183
- *Enabling Package Logging* on page 183

Selecting a Security Configuration for a Package

Designate a security configuration for a package in Sybase Control Center. This is a required step during package deployment, but you can later change the security configuration.

The administrator must create a security configuration in the cluster and assign it to the domain where the package is deployed before the deployer can assign the security configuration to the package.

1. In the left navigation pane, expand the **Packages** folder, and select the package to configure.
2. In the right administration pane, click the **Settings** tab.
3. Select a security configuration.

   The security profiles that appear in this list have been created by a platform administrator and assigned to the domain.
4. Click **Save**.

See also

- *Deploying an MBO Package to a Domain* on page 182
- *Mapping Roles for a Package* on page 183
- *Enabling Package Logging* on page 183

Mapping Roles for a Package

Configure package role mapping to authorize client requests to access MBOs and operations. Domain administrators can use a role mappings table to manage logical roles at the package level.

Note: If a developer has defined a logical role, mapping is not required; the logical role is matched to the physical role of the same name and is therefore automatically mapped.

1. Open Sybase Control Center.
2. Select and deploy an available package.
3. Follow the wizard prompts until you reach the Configure Role Mapping page for the target package. Alternately, if you are editing package role mapping after deployment, select the **Role Mapping** tab from the **Packages** > < ***PackageName***> node in the left navigation pane.
4. Set an appropriate mapping state for each logical role. The state you choose allows you to disable logical roles (**NONE**), allow logical roles to be dynamically mapped (**AUTO)**, or manually define which physical roles must be mapped to one or more logical roles (**Map Roles**). The states of AUTO or NONE require the least administration.

See also

- *Roles and Mappings* on page 112
- *Deploying an MBO Package to a Domain* on page 182
- *Selecting a Security Configuration for a Package* on page 182
- *Enabling Package Logging* on page 183

Enabling Package Logging

Information, errors and events can be recorded for packages.

Prerequisites

Package log data is sent to the domain log, only if you enable domain logging. Ensure you enable domain logging before enabling package logging.

Task

1. In the left navigation pane of Sybase Control Center, expand the **Packages** folder and select the package to configure.
2. In the right administration pane, click the **Settings** tab.
3. Enable package logging or synchronization tracing as required.
4. Click **Save**.

View, search, and export package log data from the domain log. See *Sybase Control Center online help > Manage > Managing Unwired Platform > Routine System Maintenance Tasks > Checking the Domain Log*.

See also

- *Deploying an MBO Package to a Domain* on page 182
- *Selecting a Security Configuration for a Package* on page 182
- *Mapping Roles for a Package* on page 183

Mobile Workflow Package Administration Overview

The goal of mobile workflow package management is to make mobile workflows available from the Unwired Server to device users. Mobile workflow package management typically requires a one-time deployment and configuration, except for ongoing package maintenance.

The mobile workflow application is a simple business process application that delivers functionality, such as sending requests and approvals through an e-mail application, to mobile device clients on supported device platforms, including Windows Mobile, iOS.

Table 12. Mobile workflow package management

Task	Frequency	Accomplish by using
Deploy mobile workflow packages	Once, unless a new version becomes available	Sybase Control Center for Unwired Platform with the Workflow node
Mobile workflow configuration that includes e-mail matching rules and context variables	Once	Sybase Control Center for Unwired Platform with the Workflow node
Device registration and user assignments to mobile workflow packages	Routine when new users or new devices are added	Sybase Control Center for Unwired Platform with the **Workflow** > **<WorkflowName>** node
Monitor users and errors	Routine	Sybase Control Center for Unwired Platform with the Monitor node

Enabling and Configuring the Notification Mailbox

Configure the notification mailbox settings that allow Unwired Server to transform e-mail messages into mobile workflows.

The notification mailbox configuration uses a listener to scan all incoming e-mail messages delivered to the particular inbox specified during configuration. When the listener identifies

an e-mail message that matches the rules specified by the administrator, it sends the message as a mobile workflow to the device that matches the rule.

Note: Saving changes to the notification mailbox configuration deletes all e-mail messages from the account. Before proceeding with configuration changes, consult your e-mail administrator if you want to back up the existing messages in the configured account.

1. Log in to Sybase Control Center.
2. In the left navigation pane, click **Workflows**.
3. In the right administration pane, click **Notification Mailbox**.
4. Select **Enable**.
5. Configure these properties:
   - **Protocol** – choose between POP3 or IMAP, depending on the e-mail server used.
   - **Use SSL** – encrypt the connection between Unwired Server and the e-mail server in your environment.
   - **Server** and **Port** – configure these connection properties so Unwired Server can connect to the e-mail server in your environment. The defaults are localhost and port 110 (unencrypted) or 995 (encrypted).
   - **User name** and **Password** – configure these login properties so Unwired Server can log in with a valid e-mail user identity.
   - **Truncation limit** – specify the maximum number of characters taken from the body text of the original e-mail message, and downloaded to the client during synchronization. If the body exceeds this number of characters, the listener truncates the body text to the number of specified characters before distributing it. The default is 5000 characters.
   - **Poll seconds** – the number of seconds the listener sleeps between polls. During each poll, the listener checks the master inbox for new e-mail messages to process. The default is 60 seconds.
6. If you have added at least one distribution rule, you can click **Test** to test your configuration. If the test is successful, click **Save**.

Deploying and Managing Mobile Workflow Packages

Use Sybase Control Center to deploy mobile workflow packages created by developers, and to perform the configuration tasks required to make them available to application users on messaging devices.

Multiple tasks are involved in workflow package deployment and management.

Configuring Mobile Workflow Package Properties

Use Sybase Control Center to configure the deployed mobile workflow package general properties, matching rules, and context variables to give a mobile workflow package a different set of properties in the production environment.

1. Configure general mobile workflow properties in the **General** tab of the **Workflows > MyWorkFlow** node. General mobile workflow properties include the display name and icon for the mobile workflow package.

   See *Configuring General Mobile Workflow Properties* in Sybase Control Center online help.
2. Configure and test matching rules for the package in the **Matching Rules** tab of the **Workflows > MyWorkFlow** node. Matching rules specify how to redirect e-mail messages at runtime.

   See *Configuring Matching Rules* in Sybase Control Center online help.
3. Configure context variables for the package in the **Context Variables** tab of the **Workflows > MyWorkFlow** node. Context variables specify how to load data into the Unwired Server cache as well as the domain where the MBO package is deployed.

   See *Configuring Context Variables* in Sybase Control Center online help.

See also

- *Assigning and Unassigning Device Users* on page 186
- *Deploying a Mobile Workflow Package to a Domain* on page 187

Assigning and Unassigning Device Users

Assign mobile workflow packages to make them available to a device user. Unassign them when a package is no longer required.

1. In the left navigation pane of Sybase Contorl Center, click **Workflows > MyWorkFlow**.
2. In the right administration pane, click the **Devices** tab.
3. Locate the device to assign a mobile workflow package to, then:
   a) Click **Assign Workflow**.
   b) List the activation users to assign the mobile workflow package to.

      By default, no users are listed in this window. Search for users by selecting the user property you want to search on, then selecting the string to match against. Click **Go** to display the users.
   c) Click **OK**.
4. To unassign a mobile workflow package, select the Activation User Name and click **Unassign Workflow**.

See also

- *Configuring Mobile Workflow Package Properties* on page 186
- *Deploying a Mobile Workflow Package to a Domain* on page 187

Deploying a Mobile Workflow Package to a Domain

Use Sybase Control Center to deploy a mobile workflow package to make it available on Unwired Server. Packages are initially created by developers, but are deployed and maintained on a production Unwired Server by administrators.

In the left navigation pane, from the **Workflows** node, launch the Deploy wizard for mobile workflow packages. See *Deploying a Mobile Workflow Package* in Sybase Control Center online help.

See also

- *Configuring Mobile Workflow Package Properties* on page 186
- *Assigning and Unassigning Device Users* on page 186

Managing Deployed Package Subscriptions

Manage replication, messaging, and SAP Data Orchestration Engine connector (DOE-C) package subscriptions that specify the synchronization messages mobile device users receive.

Subscription management tasks include pinging, unsubscribing, recovering, suspending, resuming, resynchronizing, and logging subscriptions. Subscription tasks vary by the package type.

These subscription management tasks apply only to the package types specified in the table below. Perform each task in the Subscriptions tab of the deployed package you are managing.

Table 13. Subscription management tasks

Subscription task	Description	Summary	Package type
Ping	Ensure that push information a user provides for a device is configured correctly. If the ping is successful, notifications and subsequent data synchronizations occur as defined by each subscription. If the ping fails, open the log and check for an incorrect host name or port number.	Select the box adjacent to the device ID, and click **Ping**.	Replication

Subscription task	Description	Summary	Package type
Unsubscribe	Remove a subscription from Unwired Server.	Select the box adjacent to the device ID, and click **Unsubscribe** for replication packages, messaging packages, and DOE-C packages. For Windows Mobile, the device application must include the DatabaseClass.CleanAllData(); method for data to be unsubscribed correctly. If this method is not used, **Unsubscribe** and **Subscribe** could work unpredictably.	All
Recover	Reestablish a relationship between the device and Unwired Server. Perform recovery under severe circumstances when a device is unable to successfully synchronize data. During subscription recovery, Unwired Server purges all enterprise data on the device. It retains the device ID and subscription information so that all data can then be resynchronized and loaded onto the device.	Check the box adjacent to the subscription ID of the device, and click **Recover**.	Messaging
Suspend/resume	Control the deactivation and reactivation of package subscriptions: • Suspend – temporarily block data synchronization for a device subscribed to a particular package. • Resume – reactivate a package subscription after it has been suspended.	Select the box adjacent to the subscription ID of the device, and click either **Suspend** or **Resume**.	Messaging DOE-C

Subscription task	Description	Summary	Package type
Resynchronize	Reactivate subscriptions to a deployed package. If a DOE-C subscription does not respond to the SAP DOE quickly enough, the DOE may mark that subscription's queues as "blocked" and stop sending messages to the DOE-C. Resynchronize to resume communication from the DOE to the DOE-C subscription.	Check the box adjacent to the subscription ID of the device, and click **ReSync**.	DOE-C
Purge	Removes subscriptions that are no longer referenced by any active users.	Select the subscription, click **Purge**, and then select the criteria.	Messaging Replication

Data Management Overview

The goal of data management is to ensure the data tier of Unwired Platform remains stable, available, and efficient. Data management is a not a routine administration task, and for Unwired Platform, primarily involves maintaining cache data and ensuring data is delivered to the device in a timely manner, as determined by your business requirements.

Table 14. Data Management Tasks

Task	Frequency	Accomplish by using
Install a new or configure an existing consolidated database (CDB) for Unwired Platform.	Once	If you are installing a new CDB, no action is required. Otherwise, use either the installer, or Sybase Control Center (postinstallation) to configure the existing database to use as the CDB. For details about using Sybase Control Center, see *Sybase Control Center online help > Configure > Configure Unwired Platform > Unwired Server > Consolidated Database*.

Task	Frequency	Accomplish by using
Set up the cache group refresh schedule (to update data on the Unwired Server from the data source).	Once, unless data refreshes need to be tuned	Sybase Control Center
Create sync paradigms (for messaging-based sync packages), synchronization groups, and device notifications (for replication-based sync packages) to customize how updated data in the cache is delivered to the device user.	Periodic, as required	Sybase Control Center
Review current/historical/performance metrics	Routine	Sybase Control Center

See also

- *Data Tier Clusters* on page 11
- *Backup and Recovery* on page 264

Data Mobility Configuration Dependencies

To ensure that data in the Unwired Platform mobility ecosystem remains effective and timely, administrators must carefully schedule and configure data update and delivery (synchronization) mechanisms in Sybase Control Center.

Consider that other dependencies might exist for the properties in the table below. Implement the properties to support both sides of the data mobility architecture (back-end to cache, and cache to frontline). In particular, poorly timed schedules or intervals can result in stagnant data or an unexpected user experience.

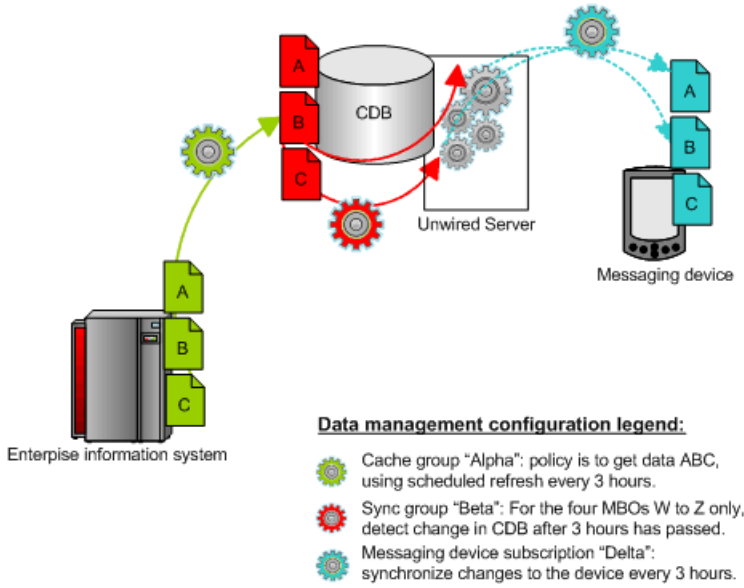
Setting	Sync type	Data updated	Usage
DCNs	Both	Server cache	The specific data units the EIS sends to Unwired Server cache. DCN updates operate outside of the cache group and cache refresh mechanisms. **Dependencies:** None. No coordination required. However, data change notifications (DCNs) do trigger the same synchronization activity on the device.
Cache group	Both	Server cache	The data the Unwired Server fetches from the EIS to update the server cache. Cache groups aggregate data updates. **Dependencies:**Design cache groups along with synchronization groups so you know which MBOs require which groups of data.

Setting	Sync type	Data updated	Usage
Cache refresh	Both	Server cache	Whether data is fetched from the EIS on demand or as scheduled. **Dependencies:**Coordinate with the synchronization group and device notifications (for replication only).
Subscriptions	Both	MBO data	The data required by the MBO. **Dependencies:**Design cache groups along with synchronization groups so you know which MBOs require which groups of data.
Sync group	MBS	MBO data	The required unit of synchronization, and the frequency with which the changes are pushed to the device via the change detection interval. **Dependencies:**Coordinate the change detection interval with the cache refresh schedule used. Device push settings also influence frequency of messages are sent.
Sync group	RBS	MBO data	The required unit of synchronization. Operation replays are sent to the server (from device), and MBOs data is then downloaded to the client. **Dependencies:**The synchronization group for replication depends on the subscription, and the device notification interval.
Device notification	RBS	MBO data	When to trigger a synchronization by sending a notification e-mail message to the device user. **Dependencies:**Coordinate the device notification interval with the cache refresh schedule.

Message Data Flow and Dependencies

Implement the properties that determine how data flows in a messaging-based synchronization paradigm in a coordinated manner.

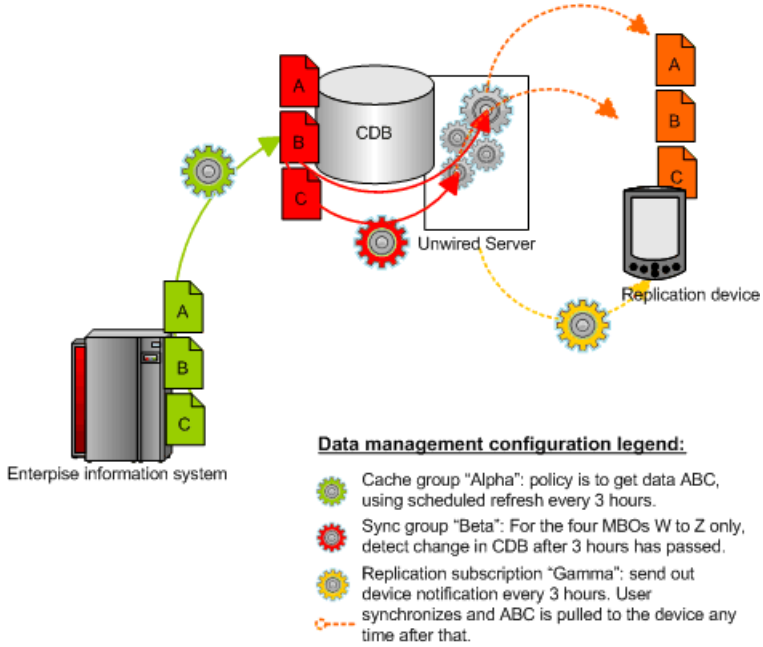
Cache refresh schedules, combined device settings and synchronization groups, work together to determine how data is disseminated to the front line from back-end EIS data sources. Coordinate these properties accordingly.



Replication Data Flow and Dependencies

Implement, in a coordinated manner, the properties that determine how data flows in a replication-based synchronization paradigm.

Cache groups, cache refresh schedules, subscriptions (in particular, synchronization groups and device notifications) work together to determine how data is disseminated to the front line from back-end EIS data sources. Coordinate these properties accordingly.



Push Synchronization for Replication Packages

If you require push synchronization (that is, synchronization initiated by Unwired Server), ensure that it is enabled and correctly set up to use specific gateways.

A gateway provides the interface that enables the notifier to send messages to a listener when MBO data changes in the Unwired Server consolidated database. Unwired Platform supports these gateways:

- HTTP – is a push-based notification for BlackBerry devices. You must manually configure this gateway.

  Note: Due to network connectivity limitations in General Packet Radio Service (GPRS)-capable Windows Mobile devices, Sybase recommends that you do not use IP number tracking and HTTP push notifications. However, because the MobiLink-based lightweight processor used for Windows Mobile devices is currently unavailable for BlackBerry, HTTP is the only gateway available for BlackBerry push. To use an HTTP gateway, you must pair the device with a BlackBerry Enterprise Server.

- DeviceTracker – is a pull-based notification for Windows Mobile devices. Uses lightweight polling using a notifier that is configured to listen for events. You do not see this gateway on the Unwired Server configuration Push Listener tab in Sybase Control Center because the gateway is configured automatically for Unwired Server. It is enabled by default and ready to use without administrator intervention.

Enabling Push and Pull Notifications

Configure push and pull for replication-based synchronization notifications.

Prerequisites

Determine the type of push synchronization gateway you require.

Task

This task sets the lightweight polling notification configuration for all clients. You can also configure it individually for each device.

1. Open and log in to Sybase Control Center.
2. In the left navigation pane, expand the **Servers** folder and select the server to configure.
3. Select **Server Configuration**.
4. In the right administration pane, click the **Replication** tab.
5. Select **Notification Configuration** from the menu bar.
6. To enable server-initiated push notification:
   a) Select **Enable Push-Based Notification**.

      Unselecting this option disables push synchronization and all related configuration properties.
   b) Enter a time interval in seconds, minutes, or hours to specify the frequency with which the push notification table is polled. The default is 10 seconds.
7. To enable client-initiated push notification:
   a) Select **Enable Pull-Based Notification**.

      Unselecting this option disables pull synchronization and all related configuration properties.
   b) Enter a time interval in seconds, minutes, or hours to specify the frequency with which the pull notification table is polled. The default is 10 seconds.
8. In the Notification Configuration menu, click **Save**.

Setting Up Lightweight Polling for a Single Client

If you do not want to set the lightweight polling configuration on Unwired Server for multiple device clients, use the client-side program for a single client.

The **poll_every** unit should be set to seconds (not minutes, hours, or a combination of the two). The lightweight poller listener on the client can be turned on/off if you do not want to receive notifications during a specific period; do not just change the interval.

1. In your program, look for where you specify the polling option right after: `...;poll_notifier=UALIGHTWEIGHT;poll_key....`

2. Change the polling option, for example: `....;poll_notifier=UALIGHTWEIGHT;poll_every=180;poll_key=.....`

Note: Do not set the client **poll_every** value to a shorter time interval than the server value. This does not result in receiving push notifications any faster, and can cause the client to see the same notification multiple times, causing multiple useless synchronizations. Only set this value on the client if for some reason you do not want to see notifications as frequently as the server checks for pending notifications.

Cache Data Management

Managing the Unwired Server cache in the consolidated database ensures that enterprise data remains stable, available, and consistent across sources. Cache data management is essential to keeping data synchronized between Unwired Server and the device client application.

Cache data consists of a copy of enterprise data that is stored in a specific area of the consolidated database (CDB). It is used as the data repository for replication and messaging mobile business objects (MBOs) that are deployed to Unwired Server. Cache management primarily involves configuring:

- Data change notifications (DCNs) -- when data used by an application changes on an EIS server, DCNs notify Unwired Server to synchronize cache data. DCNs are useful to cache management, since they facilitate time-sensitive data synchronization between scheduled cache refreshes.
- Cache refresh schedules -- update the cache with the most recent EIS data at regularly scheduled intervals, or on demand. The remote client database eventually retrieves updated data from the server's local copy in the CDB, using one of the supported synchronization triggers. Cache refresh schedules ensure data availability while managing the network traffic required to maintain that data.

Data Change Notifications

Data change notifications (DCNs) notify Unwired Server when data used by an application changes on an EIS server.

Developers typically use DCNs because they:

- Avoid the need to repopulate all the data in the mobile business object cache data every time data changes in the back-end EIS system.
- Allow the Unwired Server cache to be updated in real time by the EIS server, rather than administrator configured schedule.

DCNs are typically used in combination with a sychronization group. Part of the synchronization group is a change detection interval property. For messaging based sync applications, the property is used to send a "data change" message to the messaging based sync clients that subscribe to that data. Replication-based sync applications requires a subscription and the configuration of a device push notifications which then notifies the mobile clients.

DCNs and either data change messages or device notifications address these concerns:

- Users do not always know when to synchronize. Waiting for a user-triggered synchronization is unreliable.
- Using server-triggered push synchronization is reliable, but frequent synchronizations (for example, every X minutes) generates high traffic volumes for Unwired Server.
- Data consistency between the source EIS data and Unwired Server cache is limited to what administrators typically configure for a cache refresh interval for the entire system. It does not give priority to time-critical data.

You can configure either secure (HTTPS) or non-secure (HTTP) for DCNs. For details on how DCNs are created and sent see the *Server API Cookbook > Server API > Data Change Notification Interface*.

See also

- *Encrypting DCN Connections* on page 98
- *Configuring Unwired Server Administration Certificates* on page 85

Cache Refreshes

Cache refreshes update cache data, either on demand, or at scheduled intervals, with the most recent enterprise information.

Cache refreshes update data for mobile business objects (MBOs) that belong to the cache group undergoing the refresh. The remote client database eventually retrieves the updated data from the server's local copy in the CDB, using one of the supported synchronization triggers.

Scheduled cache refreshes control the frequency with which objects update data and regulate network traffic by synchronizing data at strategic times, rather than pushing data changes through as they occur. Administrators can also perform on-demand cache refreshes for cache groups at a specified time.

When a refresh occurs, the Unwired Server calls the default read operation (for each MBO in the cache group), and all of the rows that are returned from the enterprise information system (EIS) are compared to existing rows in the CDB as follows:

- If the CDB is empty, all rows are inserted.
- Unwired Server processes the row set and checks (using the primary key) whether the row already exists in the cache:
  - If it does, and all columns are the same as the EIS, nothing happens. When a client requests (by synchronizing) all rows that have changed since the last synchronization, only rows that have changed are included, which is important for performance and efficiency.
  - If the row does not exist, it is inserted and the next synchronization query retrieves the row.

Aggregate Updates to Multiple MBOs

Cache groups and synchronization groups control the synchronization schedule for cache and client MBO data.

Cache groups specify data refresh behavior for every mobile business object (MBO) within a group. MBOs can belong to only one cache group. Cache groups are created during development, at which point MBOs are grouped together based on their data refresh requirements. Since all MBOs in a cache group are refreshed simultaneously according to the group's cache policy, cache groups ensure that MBOs with related content remain consistent.

Synchronization groups are collections of MBOs that are synchronized together on client devices. MBOs within the same synchronization group can come from one or more cache groups. When a client synchronization request for a synchronization group is received, MBOs belonging to on-demand cache groups are refreshed according to the cache policy for that group.

Purging a Cache Group

Physically delete data that has been logically deleted from the cache. Cached data is marked as logically deleted when certain activities occur in the client application or back end.

1. In the left navigation pane of Sybase Control Center, expand the **Packages** folder and select the package to configure.
2. In the right administration pane, select the **Cache Group** tab.
3. Click **OK**.

Schedules to Manage Update Frequency

The Unwired Server cache is refreshed according to a cache policy, which defines the cache refresh behavior and properties for the MBOs within the cache group.

There are two properties that determine the cache refresh schedule, which is used with a subscription to synchronize data for mobile business objects (MBOs).

Note: You can configure a schedule refresh for a cache only if the developer specifies the cache group as a "scheduled" type during development. Otherwise, the **Schedule** tab does not appear in the Cache Properties window.

- Cache Interval – Unwired Server keeps a local copy of enterprise data in the consolidated database (CDB), and uses a synchronization server to manage updates between the CDB and EIS servers. When data is updated, the remote client database eventually retrieves updated data from this local copy in the CDB. The caching mechanism allows MBOs to retrieve updated data even if back-end servers fail. The cache interval balances the frequency with which the object updates enterprise data with the amount of network traffic required to maintain that data.

  A higher value for the cache may retain stale data, however, a lower value increases network traffic and may impede the client application's performance, because Unwired Server queries back-end information servers more frequently to look for changes and

possibly update the CDB copy. Frequent queries typically put a higher load on the servers, as well as use more network bandwidth on the server side—the cache interval does not affect required bandwidth between the synchronization server and device client applications, nor the performance characteristics of the client applications. But the interval you choose can delay synchronization if Unwired Server must first update many records in the CDB.

For example, if the cache interval is 0, each time a client application synchronizes, there is a pause while the Unwired Server rereads data from the EIS and updates the CDB. If, however, the cache interval is greater than 0, then the wait time depends on how long ago the data was refreshed. If the synchronization falls within a recent cache update, synchronization is almost immediate.

- Schedule Repeat – data is refreshed according to a schedule you set up. If you set up a schedule to repeatedly refresh data, information is always refreshed, regardless of the cache interval value. However, typically, you would set a schedule to closely match the cache interval, which is especially good practice for data that changes infrequently or is shared by many clients.

  However, as an administrator, you may use a scheduled repeat to look for data changes and to notify subscribed clients to synchronize when there are changes. Typically, mobile workers use an Unwired Platform client application as needed. To ensure that mobile data is relatively up-to-date, use the schedule repeat property to trigger a push notification.

Notifications to Update Client Data

Unwired Server alerts device users to updated mobile business object (MBO) data based on synchronization group and subscription settings. A synchronization group is a collection of MBOs that are synchronized together at regular intervals. When MBOs in a synchronization group are updated, users subscribed to those synchronization groups receive either a data update or device notification, depending on the type of application they use.

Messaging-based synchronization (MBS) clients directly receive the unit of changed data for MBOs that belong to the synchronization group. The change detection interval for the synchronization group determines how often MBS applications push MBO data to the client when cache refreshes occur.

Replication-based synchronization (RBS) clients receive device notifications when data changes are detected for any of the MBOs in the synchronization group to which they are subscribed. Both the change detection interval of the synchronization group and the notification threshold of the subscription determine how often RBS clients receive device notifications. Data updates through device notifications occur as follows:

1. Unwired Server checks the cache for data updates to MBOs according to the change detection interval configured for a synchronization group.
2. If there are data changes, the server generates device notifications.
3. Once the RBS subscription notification threshold expires, Unwired Server delivers the device notifications to clients subscribed to the synchronization group.

4. Clients receive the device notifications and synchronize data for the MBOs belonging to the synchronization group.

Administrators can use subscription templates to specify the notification threshold for a particular synchronization group. They can then use these templates to create subscriptions for device users.

For RBS devices, subscription settings ultimately determine when device notifications are delivered. For example, if data for a synchronization group is updated every two hours, but a device user's subscription indicates a notification threshold of three hours, Unwired Server postpones delivering these updates until the time indicated by the subscription settings.

Determining the Cache Interval and Schedule Refresh for a Cache Group

Use Sybase Control Center to create and configure cache groups for replication-based synchronization packages. A cache group specifies the data refresh behavior for every mobile business object (MBO) within that group.

The Unwired Server cache (also called the consolidated database, or CDB) is refreshed according to a cache policy, which defines the cache refresh behavior and properties for the MBOs within the cache group based on a schedule, or on demand.

1. Choose the cache group:
   a) In the left navigation pane, expand the **Packages** folder, and select the package.
   b) In the right administration pane, click the **Cache Group** tab.
   c) Select the cache and click **Properties**.
2. Choose an appropriate cache interval. The cache allows you to associate a frequency interval (hour, minute, seconds, and so on) for the cache group's refresh schedule. For example, if an interval value is set to 0 seconds, then every client synchronization repopulates the entire cache. Instead, choose a value that is longer in duration.
3. Set the repeat for the cycle of intervals.

   Sybase recommends that you choose a cache interval value that supports DCN behavior. See *Sybase Control Center Help > Configure > Configure Unwired Platform > Packages > Configuring a Cache Group > Configuring Scheduled Cache Groups Properties*.

Example Data Update Models

Review these scenarios to understand different data update models you can employ.

Scenario 1: Product Sales with Expected and Unexpected Changes

Consider a mobile sales team with MBOs that interact with a shared data source for product data: a Catalog MBO, a Customer MBO, and a SalesOrders MBO.

To support this deployment the administrator might:

1. Evaluate the business context.

   Of all types of data (customer, orders, product), generally product data remains the most static: the EIS server that warehouses product data is typically updated at night. At this

time, products may get added or removed, or descriptions or prices may change depending on supply and demand of those products. Regular business hours of operation are defined as 8:00 a.m. – 6:00 p.m., Monday to Friday.

2. Configure data refresh schedules based on this context.

   To accommodate the business requirements, the administrator creates a schedule repeat for the Product MBO that refreshes data daily from Monday to Friday, starting at 8:00 a.m., and a cache interval of 24 hours is used.

   The result of this configuration is that five days a week at 8:AM, the schedule *completely* reloads all Product data into the CDB, regardless of whether the CDB data is still current with respect to the 24-hour cache interval. The sales team, who knows that the data is refreshed each morning, synchronizes the Catalog MBO to update data in their local Product table.

3. Anticipate unexpected events and modify the schedule or the cache interval as required.

   Consider an incorrect price in the EIS database. If a T-shirt with a wholesale price of $10.47 is entered erroneously as $1.47, the device's Product table will be updated with the incorrect price information and must be corrected. An associate updates the EIS database, but the team must be informed of this critical data change.

   As a result, the administrator uses Sybase Control Center for Unwired Server to refresh the MBO once manually.

See also

- *Scenario 2: Urgent Alerts using Subscriptions and Schedules* on page 200

Scenario 2: Urgent Alerts using Subscriptions and Schedules

Consider an hospital's UrgentAlert MBO that has these attributes: Username, AlertTime, Message.

To support this deployment, an administrator might:

1. Evaluate the business context.

   A hospital executive demands that alerts be delivered to a specific doctor (or medical team) according to each person's unique user name, within one minute of the message being created in the EIS server.

2. Configure data refresh schedules based on this context.

   The Unwired Server administrator responds in such a way that when a new alert is entered into the back-end server, the alert is delivered to the corresponding doctor by setting the cache interval to 0 or "real time". Data is always live and immediate with no caching involved. However, the administrator also sets the schedule repeat for 1 minute and is required 24 hours a day, seven days a week. This refresh schedule is paired with a subscription template for the UrgentAlert MBO with push synchronization enabled.

When any mobile client with an application that includes the UrgentAlert MBO initially synchronizes, Unwired Server creates a push subscription. Because the administrator configured a schedule repeat for every minute, changes are delivered as they occur: when a change is detected, Unwired Server scans through the subscriptions to find all clients that are

subscribed to this MBO and determines where the Username matches. When a match occurs, Unwired Server sends the client a push notification, telling doctor to run another synchronization to retrieve the new message using the UrgentAlert MBO.

See also

- *Scenario 1: Product Sales with Expected and Unexpected Changes* on page 199

CHAPTER 10 Device User Management

Device user management applies to broad category of functions: registering devices and managing user accounts. It can also extend to activating and provisioning mobile devices.

Mobile management features of Unwired Platform allow mobile devices and their user accounts to be managed in similar fashion to how computer applications are managed by IT departments: users need to be identifiable, computers need to be tracked and assigned, and "standard" enterprise applications need to be distributed and installed.

In Unwired Platform, the device and user management paradigm includes:

- Device user management
- Device activation and setup
- Subscription services

Device and User Management Overview

The goal of device user management is to register, activate, and provision devices so that device users can access applications and be tracked and managed by the system. This is particularly important for messaging-based synchronization (MBS) environments, because users cannot access the application unless the device is registered and activated.

Table 15. Device and user management tasks

Task	Frequency	Accomplish by using
Activate and deactivate application devices according to the type of synchronization model used (RBS or MBS)	When a new user of a messaging or mobile workflow package must be added	Sybase Control Center for Unwired Platform with the Device Users node.
Review registered devices and users, delete devices to free licenses, delete users to remove them from the system	As required	Sybase Control Center for Unwired Platform with the Device Users node.
Manage subscriptions	As required	Sybase Control Center for Unwired Platform with the Packages node.
Add and manage device templates	As required	Sybase Control Center for Unwired Platform with the Device Users node.

Users

In Unwired Platform, application users are individuals who have been registered through messaging-based or replication-based applications. Application users are managed in Sybase Control Center.

A user is automatically registered upon first successful authentication by the security provider associated with the package that user is trying to access from a device. A user can have multiple devices.

The platform administrator can view the user name and the security configuration that was used to authenticate the user. The administrator can also review devices associated with a user to perform any device deletion to free up license. In addition, the administrator can remove users that no longer exist.

Note: SAP DOE-C package users are not registered in Unwired Server. Those users are authenticated by their respective DOE back-end servers.

Messaging Devices

Messaging devices contain applications that send and receive data through messaging. A platform administrator must configure the device activation template properties for messaging-based synchronization (MBS) devices. Device activation requires user registration. Upon successful registration, the device is activated and set up with the template the administrator has selected.

Device registration pairs a user and a device once the user supplies the correct activation code. This information is stored in the messaging database, which contains extensive information about users and their corresponding mobile devices.

Users who are registered but who have not yet installed the software are listed in the window as **registered**, and their messages are queued by Unwired Server for later delivery.

Typically, device registration occurs when the user initially attempts to connect to Unwired Server. However, an administrator can force a user to reregister if there is data corruption on the device, or if the user is assigned a new device. This reestablishes the relationship between the server and the device and refreshes the entire data set on the device.

iOS devices are listed with other messaging device types. However, unlike other device types, each application on an iOS device has its own entry that concatenates the device id with the application shortname.

Note: The device user must activate the messaging account within the number of hours specified in their activation message. If a user does not activate the account within that time frame, an administrator must reregister the user.

Device Registration and Activation

Device registration and activation is a required step for a messaging based synchronization environment. These two steps are dependent actions that make the device and the user that is associated with it, part of the Unwired Platform mobility system.

Device registration begins with the registration notice and device template you create once and then reuse in Sybase Control Center. The registration notifies users that they must identify themselves by pairing the user identity and device identity. The device template then sets up the device and makes it compatible for Unwired Platform messaging use. You can use as many templates as you required.

Once the registration notification is received, the user submits the information back to Unwired Server. This action creates a registration slot that gets occupied by the device upon device activation.

Once the device is activated, the process then opens the communication channel used to relay messages and events to the device. A user can create subscriptions which are then associated with the registered device.

MBS Device Maintenance

As changes occur to the users in your messaging environment, you need to change who is eligible to access data remotely and thereby modify the device state over time.

Use this table to help you assess what administration action to perform from the Sybase Control Center Device Users node when the status of a user or device changes:

Scenario	Device management feature required
A user in the system loses, breaks, or receives a new device.	The **Devices** tab allows you to clone information for MBS devices. Cloning creates a duplicate copy of a device user's messaging configuration settings. This duplication allows you to retain user information but instead pair it with a different device. The administrator can also lock or delete the device, as required.
Synchronization fails and you need to: • Reestablish a relationship between the device and Unwired Server. • Purge existing data from the device. The user fails to register within the allocated time period.	The **Devices** tab allows you to reregister devices. During device reregistration, Unwired Server purges all enterprise data on the device. It retains the device ID and subscription information so that all data can then be resynchronized and loaded onto the device.

Scenario	Device management feature required
A user has multiple devices associated with them, and you need to view those devices.	The User node **Users** tab allows you to open a View Devices dialog. All devices paired with that user ID are listed there.
A device has changed ownership from one user to another.	The User node **Devices** tab allows you to list all devices registered and active with Unwired Server. The administrator can reregister the device and delete the old user.
A user no longer works for your organization or has taken an extended leave. Irrespective of whether the device belongs to the organization or the user, you want this account disabled and thereby not allow the user to access data remotely any longer.	The **Devices** tab allows you disable the entire device account. When you delete a device, all corresponding package subscriptions related to that device are removed from the system, and the license used by that device is returned to the pool of available license slots tracked by Unwired Server. Actual user data, such as personalization keys, is not deleted.
A support request requires an end-to-end investigation of the environment. You need to access the device log in addition to server environment logs.	The **Devices** tab allows you to send a request to Unwired Server to retrieve log files from one or more messaging-based synchronization (MBS) devices.

For details about all of these actions and how they are performed, see *Sybase Control Center online help>Manage>Managing Unwired Platform>Routine Command and Control Actions>Provision>Device Users>Devices*.

Replication Devices

Replication devices are used with replication-based synchronization (RBS) mobile business objects that rely on RBS data cached in the consolidated database. RBS device users are automatically registered when they first synchronize data. There is no device configuration required; the only tasks an administrator performs are monitoring RBS device activity, locking and unlocking RBS devices, and deleting them.

An administrator can lock or unlock devices to disallow or allow users from the device to access the Unwired cluster. All activities, including sending of the push notifications, are stopped while a device is locked. The device application will get an error when trying to communicate with Unwired Server. See *Sybase Control Center > Manage > Managing Unwired Platform > Routine Command and Control Actions > Provision > Device Users > Devices > Locking and Unlocking Devices*.

RBS Device Maintenance

As changes occur to the users in your replication environment, you need to change who eligible to access data remotely and thereby modify the device state over time.

Use this table to help you assess what administration action to perform from the Sybase Control Center Device User node when the status of a user or device changes:

Scenario	Device management feature required
A user in the system loses, breaks, or receives a new device.	None. The registration of the device is automatic for replication environments. This registration occurs during synchronization. The administrator can lock or remove the device, as required.
A user has multiple devices associated with them, and you need to view those devices.	The Device User node **Users** tab allows you to open a View Devices dialog. All devices paired with that user ID are listed there.
A device has changed ownership from one user to another.	The Device User node **Devices** tab allows you to list all devices registered and active with Unwired Server. The administrator can delete the device and the old user from the system.
An RBS device needs to now support MBS applications in addition to RBS applications.	The Device User node **Devices** tab allows send an upgrade request that also allows the user to register the device with Unwired Server as a messaging device. You must then administer the device as both an RBS and MBS device.
A user is taking a short leave and you do not want to change the security configuration required for the package to temporarily disable access.	The **Devices** tab allows you lock and unlock the device account. Lock or unlock devices to control which users are allowed to synchronize data.

For details about all of these actions and how they are performed, see *Sybase Control Center online help>Manage>Managing Unwired Platform>Routine Command and Control Actions>Provision>Device Users>Devices*.

Subscriptions

Subscriptions are managed with MBO packages. However, they are closely associated nonetheless with the device user, because it is the pairing of a user profile and registered and activated device account give the user subscription privileges. Subscriptions define what subset of data the user wants to receive when they synchronize.

Subscription management actions, by whom the action is performed (device user or administrator), as well as the information contained in the subscription, varies depending on the application data required by the MBO package deployed.

Package type	Action	Performed by
Replication	Create subscription template to allow the device user to be notified when information is available, depending on the subsciption configured for that package.	Administrator
	View subscription information, which include the device ID, the device user, and the synchronization group notification threshold, sync counts, and last sync time.	Administrator
	Configure push notifications to activate notification delivery and set the push frequency.	Device user
Messaging (and the DOE-C subtype)	Subscribe to data to determine the frequency of data deliveries as well as other properties associated with messaging synchronization.	Device user
	View subscription information, which includes the device ID, the device user, device status, the last outbound response time, log levels,nd the application name and application client ID that uses the subscription.	Administrator

For information about subscription properties, see *Sybase Control Center online help > Configure > Configuring the Unwired Platform> Packages*.

CHAPTER 11 Runtime Monitoring

Runtime monitoring allows you to gauge the health of your mobile enterprise and troubleshoot issues that arise. Use Sybase Control Center with system logs for this purpose.

Sybase Control Center-based monitoring reduces the burden of vital, but time-consuming routine tasks, by freeing IT and administration staff to focus on other initiatives, without reducing operational health of Unwired Platform for the organization.

When used in conjunction with regular analysis of system logs, administrators can:

- Keep devices maintained and performing efficiently
- Keep components available and reduce failure length
- Identify and react to security or synchronization events before they impact your mobile infrastructure

System Monitoring Overview

The goal of monitoring is to provide a record of activities and performance statistics for various elements of the application. Monitoring is an ongoing administration task.

Use monitoring information to identify errors in the system and resolve them appropriately. This data can also be shared by platform and domain administrators by exporting and saving the data to a .CSV or .XML file.

The platform administrator uses Sybase Control Center to monitor various aspects of Unwired Platform. Monitoring information includes current activity, historical activity, and general performance during a specified time period. You can monitor these components:

- Replication-based synchronization
- Messaging-based synchronization
- System queue status
- Data change notifications
- Device notifications (RBS)
- Package statistics
- Device users
- Cache activity

To enable monitoring, platform administrators must set up a monitoring database, configure a monitoring data source or create a new one, and set up monitoring database flush and purge options. By default the installer created a monitoring database, however you can use another one if you choose.

To control monitoring, platform administrators create monitoring profiles and configurations, which define the targets (domains and packages) to monitor for a configured length of time. A

default monitoring profile is created for you by the installer. Monitoring data can be deleted by the platform administrator as needed.

Table 16. System monitoring tasks

Task	Frequency	Accomplished by
Create and enable monitoring profiles	One-time initial configuration with infrequent tuning as required	Sybase Control Center for Unwired Platform with the Monitoring node
Enable domain logging	One-time setup with infrequent configuration changes, usually as issues arise	Sybase Control Center for Unwired Platform with the **Domains** > <**DomainName**> > **Log** node.
Review current/historical/performance metrics	Routine	Sybase Control Center for Unwired Platform with the Monitoring node
Identify performance issues	Active	Sybase Control Center for Unwired Platform with the Monitoring node
Monitor application and user activity to check for irregularities	Active	Sybase Control Center for Unwired Platform with the Monitoring node
Troubleshoot irregularities	Infrequent	Reviewing various platform logs
Purge or export data	On demand	Sybase Control Center for Unwired Platform with the Monitoring node

Status and Performance Monitoring

Determine whether your environment is working efficiently and obtain detailed information about the status of the resources in your environment.

The Monitor node in Sybase Control Center is typically used to monitor:

- System diagnostics -- values in the Package, User, Replication, Messaging, and Messaging Queue tabs can help you troubleshoot the system when device clients or applications behave unexpectedly. Information is separated by the package type, either replication-based or messaging-based synchronization. Check to see if synchronization is blocked, during which synchronization phase, and identify the MBO or operation that may be causing a problem.
- Performance indicators -- key performance indicator columns, particularly in the Replication, Messaging, Data Change Notification, and Cache tabs, can help you identify

trends that can be used for tuning your system or determining when to scale up deployed server and data components.

Note: To perform robust data analytics, export information to an XML or a CSV (comma-separated value) file and use an analytic tool of your choosing. See *Sybase Control Center online help > Monitoring > Monitoring Unwired Platform*.

See also

- *Monitoring Unwired Platform* on page 211
- *Reviewing System Monitoring Data* on page 216
- *Monitoring Database Schema* on page 372
- *Configuring Monitoring Performance Properties* on page 214
- *Deploying the Monitoring Database* on page 28

Monitoring Unwired Platform

Configure settings to audit the performance and availability of server and application environments.

Monitored operations include replication-based synchronization, messaging-based synchronization, messaging queue, data change notification, device notification, package, user, and cache activity. These aspects of monitoring are important to ensuring that the required data is collected.

The critical aspects of monitoring include:

1. Setting up a monitoring configuration. A monitoring configuration sets the server behavior for writing data to database, automatic purge, and data source where the monitoring data is stored.

   A default configuration is created for you, however you will likely want to customize this configuration for your environment. By default, monitoring data is flushed every 5 minutes. In development and debugging scenarios, you may need to set the flush behavior to be immediate. Set the **Number of rows** and **Batch size** properties to a low number. You can also disable flush, which results in immediately persisting changes to monitoring database. If you are setting up immediate persistence in a production environment, you may experience degraded performance. Use persistence with caution.
2. Creating a monitoring profile. A monitoring profile defines one or more domains and packages that need to be monitored.

   You can either use the **default** profile to capture monitoring data for all packages in all domains or create specific profiles as required. Otherwise, disable the **default** profile or modify it as needed.
3. Reviewing the captured data. An administrator can review monitoring data (current, historical, and performance statistics) from Sybase Control Center.

Use the monitoring tabs to filter the data by domain, package, and time range. You can also export the data into a CSV or XML file and then use any available reporting or spreadsheet tool to analyze the data.

See also

- *Reviewing System Monitoring Data* on page 216

Monitoring Profiles

Monitoring profiles specify a monitoring schedule for a particular group of packages. These profiles let administrators collect granular data on which to base domain maintenance and configuration decisions.

A default monitoring profile is automatically created in disabled state on Unwired Server. Administrators can enable or remove the default profile, and enable one or more new monitoring profiles as required.

The same monitoring schedule can be applied to packages across different domains; similarly, you can select individual packages for a monitoring profile.

Note: Properties you configure for an Unwired Server are cluster-affecting. Therefore, to make sure they are propagated correctly, Sybase recommends that you set them only on a primary cluster server.

Planning for System Monitoring

Planning Unwired Platform performance monitoring requires performance objectives, benchmarks based on those objectives, and plans for scaling your production environment. Do this before you create your monitoring profile.

1. Understand the business requirements your system is addressing.
2. Translate those business needs into performance objectives. As business needs evolve, performance objectives must also evolve.
3. Perform capacity planning and evaluate performance of the monitoring database.

   If statistics indicate that Unwired Platform is approaching your capacity guidelines, you may want to collect this data more frequently and flush stale data. You can also use the Administration Client API to automate tasks such as exporting and purging of monitoring data.
4. Define and document a base set of statistics, which might include:
   - Peak synchronizations per hour
   - Peak synchronizations per day
   - Acceptable average response time
5. Decide how often system statistics must be monitored to create benchmarks and evaluate results for trends. Use benchmarks and trends to determine when your production

environment needs to be expanded. Trend analysis should be performed on a regular basis, perhaps monthly.

Next

Open Sybase Control Center and create and configure the profiles you require to support your planning process.

Creating and Enabling a Monitoring Profile

Specify a monitoring schedule for a group of packages.

Prerequisites

Depending on the expected level of monitoring activity, ensure that the monitoring database is adequately prepared to store monitoring data.

Task

1. Open and log in to Sybase Control Center.
2. In the left navigation pane, select **Monitoring**.
3. In the right administration pane, select the **General** tab.
4. Click **New** to create a monitoring profile.
5. Enter a name for the new profile.
6. Select the **Domains and Packages** tab and choose packages to be monitored according to these options:
   - Monitor all domains and packages – select **All Domains and Packages**.
   - Monitor all packages from one or more domains – select a domain, then click **Select All Packages**. Perform this step for each domain you want to monitor.
   - Monitor specific packages from one or more domains – select a domain, then select the particular packages you want to monitor from that domain. Perform this step for each domain you want to monitor.
7. Select **View my selections** to view the packages you selected for the monitoring profile. Unselect this option to return to the package selection table.
8. Select **Enable after creation** to enable monitoring for the selected packages immediately after you create the profile. By default, this option is selected. Unselect this option to enable the monitoring profile later.
9. On the **Schedule** tab, select a schedule to specify when monitoring takes place:
   - **Always On** – this schedule requires no settings. Package activity is continually monitored.
   - **Run Once** – specify a length of time during which monitoring occurs, in either minutes or hours. Package activity is monitored for the duration specified for one time only.

- **Custom** – specify start and end dates, start and end times, and days of the week. Package activity is monitored according to the time frame specified. See *Setting a Custom Monitoring Schedule*.

10. Click **OK**.
    A status message appears in the administration pane indicating the success or failure of profile creation. If successful, the profile appears in the monitoring profiles table.
11. To enable a profile that you did not enable during creation, select the monitoring profile and click **Enable**.

Setting a Custom Monitoring Schedule

Customize the monitoring schedule for packages within a monitoring profile in Sybase Control Center. Setting a custom schedule is the most flexible option; monitoring information is provided according to the time frame you specify.

Prerequisites

Begin creating a monitoring profile in the New Monitor Profile dialog.

Task

1. In the New Monitor Profile dialog, select the **Schedule** tab.
2. Select **Custom** as the monitoring schedule criteria.
3. To set a range to control which days the custom schedule runs, configure a start date and time, end date and time, or day of week (if applicable).

   - Select **Start Date** to set a date for when monitoring of package activity begins. To be more specific, you can also enter a **Start Time**. In this case, monitoring cannot begin until a given time on a given day has been reached.
   - Select **End Date** to set a date that ends the monitoring of package activity. To be more specific, you can also enter an **End Time**.
   - Select the days of the week that package monitoring runs. This means that for the days you select, the schedule runs every week on the day or days you specify.

   If you do not indicate a time frame, Unwired Server uses the default custom schedule, which is equivalent to Always On monitoring.
4. Click **OK**.

Configuring Monitoring Performance Properties

Configure auto-purge, flush threshold, and flush batch size settings to determine how long monitoring data is retained, and set a monitoring database to configure where data is stored.

Prerequisites

Depending on the expected level of monitoring activity, ensure that the monitoring database is adequately prepared to store monitoring data.

Task

1. In the left navigation pane of Sybase Control Center, select **Monitoring**.
2. In the right administration pane, select the **General** tab.
3. Click **Configuration**.
4. Configure auto purge settings.

   Auto purge clears obsolete data from the monitoring database once it reaches the specified threshold.

   a) Select **Enable auto purge configuration** to activate auto purge functionality.
   b) Enter the length of time (in days) to retain monitoring data before it is purged.
5. Configure flush threshold settings.

   The flush threshold indicates how often data is flushed from memory to the database. This allows you to specify the size of the data saved in memory before it is cleared. Alternately, if you do not enable a flush threshold, data is automatically written to the monitoring database as it is captured.

   a) Select **Enable flush threshold configuration** to activate flush threshold functionality.
   b) Select one of:
      - **Number of rows** – monitoring data that surpasses the specified number of rows is flushed from memory. Enter the desired number of rows adjacent to **Rows**. The default is 100.
      - **Time interval** – monitoring data older than the specified time interval is flushed from memory. Enter the desired duration adjacent to **Minutes**. The default is 5.
      - **Either rows or time interval** – monitoring data is flushed from memory according to whichever value is reached first: either the specified number of rows or the specified time interval. Enter the desired rows and duration adjacent to **Rows** and **Minutes**, respectively.
6. If you enabled a flush threshold, enter a **Flush batch row size** by specifying the size of each batch of data sent to the monitoring database. The row size must be a positive integer.

   The batch size divides flushed data into smaller segments, rather than saving all data together according to the flush threshold parameters. For example, if you set the flush threshold to 100 rows and the flush batch row size to 50, once 100 rows are collected in the monitoring console, the save process executes twice; data is flushed into the database in two batches of 50 rows. If the flush threshold is not enabled, the flush batch row size is implicitly 1.

   Note: By default, the monitoring database flushes data every 5 minutes. Alternatively, you can flush data immediately by removing or decreasing the default values, but doing so impacts performance and prevents you from using captured data.
7. Optional. To change the data source, select an available database from the **Monitor database endpoint** drop down list.

   Available databases are those with a JDBC server connection type (either ASE or SQL Anywhere) created in the default domain. To create a new monitor database, a platform

administrator must set up a database by running the appropriate configuration scripts and creating a server connection for the database in the default domain. The database server connection then appears as an option in the Monitor Database Endpoint drop down list.

8. Click **OK**.

See also

- *Status and Performance Monitoring* on page 210
- *Monitoring Database Schema* on page 372

Monitoring Usage

Monitoring information reflects current and historical activity, and general performance during a specified time period.

Monitoring allows administrators to identify key areas of weakness or periods of high activity in the particular area they are monitoring. Access to this data helps administrators make decisions about how to better configure the application environment to achieve a higher level of performance.

The historical data is preserved in the monitor database. Performance data (KPIs for Replication, Messaging, Package Statistics, User Statistics, and Cache Statistics) for the specified time period is calculated upon request using the historical data available for that period. If monitoring data is purged for that time period, the performance data calculations will not factor in that data. It is recommended to purge monitoring data after putting in place mechanisms to export the required historical and/or performance data as needed. By default, monitoring data is automatically purged after seven days.

Also note that the processing times are calculated based on the time the request (or message) arrives on the server, and the time it took to process the request (or message) on the server. The client-side time (request origin time, and time taken to deliver to the server) are not factored into that data.

Reviewing System Monitoring Data

Review data for monitored activities in Sybase Control Center. The monitoring data is retrieved according to the specified time range. Key Performance Indicators (KPIs) are also calculated for the specified time range.

1. Open and log in to Sybase Control Center.
2. In the left navigation pane, select **Monitoring**.
3. In the right administration pane, select one of the following tabs according to the type of monitoring data you want to view:
   - **Replication**
   - **Messaging**
   - **Queue**

- **Data Change Notifications**
- **Device Notifications**
- **Package Statistics**
- **User Statistics**
- **Cache Statistics**

See also

- *Monitoring Unwired Platform* on page 211

Current and Historical Data

Monitoring data is categorized as current, historical, or performance.

Current data for replication MBOs, cache, and replication packages is tracked in subtabs. Use current data to assess the state of an object and check for abnormalities: for example, whether the application is working, or if the current data request is blocked.

As the request is processed, current data is moved to historical tables and used to calculate the performance data. Use accumulated history data to derive object performance statistics: each time an activity is performed on the object, activity indicators are recorded in the table and create meaningful aggregate statistics.

Performance Data: KPIs

Performance subtabs list key performance indicators (KPIs). KPIs are monitoring metrics that use counters, activities, and time measurements, that show the health of the system. KPIs can use current data or historical data.

Metrics help administrators ascertain how close an application is to corporate performance goals. In addition, they also help you identify problem areas and bottlenecks within your application or system. Performance goal categories might include:

- System metrics related to Unwired Platform components, EIS servers, networks, and throughput of all of these elements.
- Application metrics, including counters.
- Service-level metrics, which are related to your application, such as orders per second and searches per second.

When reviewing monitoring data, administrators typically start by evaluating high-level data and may then choose to drill down into object specifics. The approach used typically depends on the goal: benchmark, diagnose, or assess system health.

KPI type	Description	Used to
Counters	Counters keep a grand total of the number of times something happens, until historical data is purged.	Monitor the system workload. Performance objectives derived from the workload are often tied to a business requirement such as a travel purchasing application that must support 100 concurrent browsing users, but only 10 concurrent purchasing users.
Time	Benchmark or assess the duration of an object or activity.	Assess the response times and latency of an object to assess service levels.
Object details	Provide summary or expanded details by object name (for example, a mobile business object, a domain, or a package). This information increases the layer of detail to counters and time values, to give context to trends suggested by the first two types of KPIs.	Analyze overall system health and troubleshoot system-wide issues.

Performance Statistics

As your production environment grows in size and complexity, perform periodic performance analysis to maintain the health of your mobility system.

The Monitoring node in Sybase Control Center is a data collection tool that helps administrators identify the causes of performance problems.

Use the Monitoring node to:

- Track application performance and identify trends
- Isolate performance problems to the resource, application, client, or user

Monitoring Data Categories

Monitoring data is organized according to object type, allowing administrators to perform focused data analysis on specific activities and Unwired Platform components. Current, historical, and performance-based statistics facilitate user support, troubleshooting, and performance tracking for individual application environments.

The replication and messaging categories are the primary sources of data relating to application environment performance. The remaining tabs present detailed monitoring data that focuses on various aspects of replication-based applications, messaging-based applications, or both.

Replication Statistics

Replication statistics reflect replication-based synchronization (RBS) activity for monitored packages. Current statistics monitor the progress of real-time synchronizations, while historical statistics present data from completed synchronizations on a per-package basis. Performance monitoring uses key performance indicators to produce data about synchronization efficiency.

Through statistics that report on the duration and scope of synchronizations, as well as any errors experienced during synchronization, replication monitoring allows you to identify the rate at which synchronizations happen during specified time periods, which users synchronize data, and which mobile business objects are affected.

Current Replication Statistics

Current statistics for replication-based synchronization (RBS) provide real-time information about in-progress synchronizations.

Unwired Server monitors RBS requests using these statistical categories:

Category	Description
Package	The package name.
Phase	The current synchronization activity: upload or download. During the upload phase, a client initiates operation replays to execute mobile business object (MBO) operations on the back-end system. During the download phase, a client synchronizes with Unwired Server to receive the latest changes to an MBO from the back-end system.
Entity	During the download phase, the name of the MBO with which the client is synchronizing. During the upload phase, the name of the operation that the client is performing.
Synchronization Start Time	The date and time that the synchronization request was initiated.
Domain	The domain to which the package involved in synchronization belongs.
Device ID	The ID number of the mobile device participating in the synchronization.
User	The name of the user associated with the device ID.

Replication History Statistics

Historical data for replication-based synchronization consists of past synchronization details for monitored packages.

The summary view provides general information, whereas the detail view presents a more specific view of all request events during each synchronization; each row of data corresponds to a synchronization request from the client in the time frame you define:

- Click either **Details** to see more granular information on each synchronization request, or select the **Detail** option to see all synchronization request details. Detail view allows you to look at the individual messages that make up the summary view.
- Select **Summary** to see aggregated details by by domain, package, and user about past synchronization events for the defined time frame.

Table 17. Detail view information

Synchronization element	Description
Package	The package name.
Device ID	The ID number of the mobile device that participated in the synchronization request.
User	The user associated with the device ID.
Phase	The sync activity that occurred during this part of synchronization: upload or download. During the upload phase, a client initiates operation replays to change an MBO. During the download phase, a client synchronizes with Unwired Server to receive the latest changes to an MBO.
Entity	During download, the name of the MBO that the client is synchronizing with. During upload, the operation that the client is performing: create, update, or delete.
Total Rows Sent	The total number of rows sent during package synchronization. This data type is not supported at the MBO level.
Bytes Transferred	The amount of data transferred during the synchronization request.
Start Time	The date and time that the synchronization request was initiated.
Finish Time	The date and time that this part of synchronization completed.
Error	The incidence of errors during this request: true or false.
Domain	The domain to which the package involved in synchronization belongs.

Table 18. Summary view information

Category	Description
User	The name of the user associated with the device ID.
Package	The package name.
Total Rows Sent	The total number of rows sent during package synchronization.
Total Operation Replays	The total number of operation replays performed by clients during synchronization.
Total Bytes Sent	The total amount of data (in bytes) downloaded by clients from Unwired Server during synchronization.
Total Bytes Received	The total amount of data (in bytes) uploaded to Unwired Server by clients during synchronization.
Start Time	The date and time that the synchronization request was initiated.
Total Synchronization Time	The amount of time taken to complete the synchronization.
Total Errors	The total number of errors that occurred for the package during synchronization.
Domain	The domain to which the package involved in synchronization belongs.

Replication Performance Statistics

Replication performance statistics consist of key performance indicators (KPIs) that reflect the overall functioning of the application environment.

Performance monitoring highlights key totals and identifies average, minimum, and maximum values for primary activities. These calculations are dynamic, and are based on the data currently available in monitoring database for the specified time period.

All values in this table (totals, averages, maximums, minimums) apply to the specific time period you indicate:

KPI	Description
Total Distinct Package Synchronization	The total number of packages subject to synchronization.
Total Distinct Users	The total number of users who initiated synchronization requests. This value comprises only individual users, and does not count multiple synchronizations requested by the same user.

KPI	Description
Average/Minimum/Maximum Sync Time	The average, minimum, or maximum amount of time Unwired Server took to finish a complete synchronization.
Time at Minimum/Maximum Sync Time	The time of day at which the shortest or longest synchronization completed.
Package with Minimum/Maximum Synchronization Time	The name of the package and associated MBO with the shortest or longest synchronization time.
Average/Minimum/Maximum MBO Rows Per Synchronization	The average, minimum, or maximum number of MBO rows of data that are downloaded when synchronization completes.
Average/Minimum/Maximum Operation Replays per Sync (records received)	The average, least, or greatest number of operation replays per synchronization received by Unwired Server from a client.
Total Bytes Sent	The total number of bytes downloaded by clients from Unwired Server.
Total Bytes Received	The total number of bytes uploaded from clients to Unwired Server.
Total Operation Replays	The total number of operation replays performed on the EIS.
Total Errors	The total number of errors that took place across all synchronizations.
Average/Minimum/Maximum Concurrent Users	The average, least, or greatest number of users involved in concurrent synchronizations.
Time at Minimum/Maximum Concurrent Users	The time at which the least or greatest number of users were involved in concurrent synchronizations.

Messaging Statistics

Messaging statistics report on messaging-based synchronization (MBS) activity for monitored packages.

- Current monitoring data tracks the progress of messages from device users presently performing operation replays or synchronizing MBOs.
- Historical data reveals statistics indicating the efficiency of completed transactions.
- Performance monitoring provides an overall view of MBS activity intended to highlight areas of strength and weakness in the application environment.

Messaging historical data captures messages such as login, subscribe, import, suspend, resume and so on. The Import type message is a data payload message from server to client (outbound messages), while rest of the messages (login, subscribe, replay, suspend, resume) are sent from the client to server (inbound messages).

Current Messaging Statistics

Current statistics for messaging-based synchronization (MBS) provide real-time information about in-progress synchronizations. Because messaging synchronizations progress rapidly, there is typically little pending MBS data available at any given time.

Unwired Server monitors MBS requests using these categories:

Category	Description
Package	The package name.
Message Type	The type of message sent by the client to Unwired Server, indicating the current sync activity; for example, import, replay, subscribe, suspend, resume, and so on.
Entity	During the import process, the name of the mobile business object (MBO) with which the client is synchronizing. During replay, the operation that the client is performing. For all other message types, the cell is blank.
Start Time	The date and time that the initial message requesting synchronization was sent by the client.
Domain	The domain to which the package involved in synchronization belongs.
Device ID	The ID number of the mobile device participating in the synchronization.
User	The name of the user associated with the device ID.

Messaging History Statistics

Historical data for messaging-based synchronization consists of past synchronization details for monitored packages.

The summary view provides general information, whereas the detail view presents a more specific view of all request events during each synchronization; each row of data corresponds to a synchronization request from the client in the time frame you define:

- Click either **Details** to see more granular information on each synchronization request, or select the **Detail** option to see all synchronization request details. Detail view allows you to look at the individual messages that make up the summary view.
- Select **Summary** to see aggregated details by by domain, package, and user about past synchronization events for the defined time frame.

Table 19. Detail view information

Data type	Description
Package	The package name.
Device ID	The ID number of the mobile device that participated in the synchronization request.
User	The name of the user associated with the device ID.
Message Type	The type of message sent by the client to Unwired Server, indicating the sync activity; for example, import, replay, subscribe, suspend, resume, and so on.
Entity	During the import process, the name of the mobile business object (MBO) that the client is synchronizing with. During replay, the operation that the client is performing. For all other message types, the cell is blank.
Payload Size	The size of the message (in bytes).
Start Time	The date and time that the message for this sync request is received.
Finish Time	The date and time that the message for this sync request is processed.
Processing Time	The total amount of time between the start time and the finish time.
Error	The incidence of errors during this request; either true or false.
Domain	The domain to which the package involved in synchronization belongs.

Table 20. Summary view information

Category	Description
User	The name of the user associated with the device ID
Package	The package name
Total Messages Sent	The total number of messages sent by Unwired Server to clients during synchronization
Total Messages Received	The total number of messages received by Unwired Server from clients during synchronization
Total Payload Size Sent	The total amount of data (in bytes) downloaded by clients from Unwired Server during synchronization

Category	Description
Total Payload Size Received	The total amount of data (in bytes) uploaded to Unwired Server by clients during synchronization
Total Operation Replays	The total number of operation replays performed by clients during synchronization
Last Time In	The date and time that the last inbound request was received
Last Time Out	The date and time that the last outbound response was sent
Subscription Commands Count	The total number of subscription commands sent during synchronization; for example, subscribe, recover, suspend, and so on
Total Errors	The number of errors that occurred for the package during synchronization
Domain	The domain to which the package involved in synchronization belongs

Messaging Performance Statistics

Messaging performance statistics consist of key performance indicators (KPIs) that reflect the overall functioning of the application environment.

Performance monitoring highlights key totals and identifies average, minimum, and maximum values for primary activities. These calculations are dynamic, and are based on the data currently available in monitoring database for the specified time period.

All values in this table (totals, averages, maximums, minimums) apply to the specific time period you indicate:

KPI	Description
Total Messages	The total number of messages sent between the server and clients during synchronization.
Total Distinct Devices	The total number of devices involved in synchronization. This total includes the same user multiple times if he or she has multiple devices. The value comprises individual devices, and does not count multiple synchronizations requested by the same device.
Total Distinct Users	The total number of users who initiated synchronization requests. This value comprises individual users, and does not count multiple synchronizations requested by the same user if he or she uses multiple devices.
Average/Minimum/Maximum Concurrent Users	The average, minimum, or maximum number of users involved in simultaneous synchronizations.

KPI	Description
Time at Minimum/Maximum Concurrent Users	The time at which the greatest or least number of users were involved in concurrent synchronizations.
Average/Minimum/Maximum Processing Time	The average, minimum, or maximum amount of time Unwired Server took to respond to a sync request message.
Time at Minimum/Maximum Message Processing Time	The time of day at which the shortest or longest message processing event completed.
MBO for Maximum/Minimum Message Processing Time	The name of the package and associated mobile business object (MBO) with the shortest or longest message processing time.
Average/Minimum/Maximum Message Size	The average, smallest, or largest message sent during synchronization.
Total Inbound Messages	The total number of messages sent from clients to Unwired Server.
Total Outbound Messages	The total number of messages sent from Unwired Server to clients.
Total Operation Replays	The total number of operation replays performed by clients on MBOs.
Total Errors	The total number of errors that took place across all synchronizations.
Average/Minimum/Maximum Concurrent Users	The average, least, or greatest number of users involved in concurrent synchronizations.

Messaging Queue Statistics

Messaging queue statistics reflect the status of various messaging queues. The data does not reveal any application-specific information, but provides a historical view of messaging activities that communicates the efficiency of messaging-based synchronization, as well as the demands of device client users on the system.

Based on this data, administrators can calculate the appropriate inbound and outbound message queue counts for the system (configurable in the Server Configuration node of Sybase Control Center). See *Sybase Control Center online help > Configure > Configuring Unwired Platform > Unwired Server > Server Properties > Configuring System Performance Properties*.

Messaging Queue Status

Messaging queue status data provides historical information about the processing of messaging-based synchronization requests by Unwired Server. The data indicates areas of high load and times of greatest activity. This data can help administrators decide how to handle queue congestion and other performance issues.

These key indicators monitor messaging queue status:

Statistic	Description
Name	The name of the messaging queue.
Current Queued Items	The total number of pending messages waiting to be processed by Unwired Server.
Average/Minimum/Maximum Queue Depth	The average, minimum, or maximum number of queued messages. For minimum and maximum queue depth, this value is calculated from the last server restart.
Time at Minimum/Maximum Queue Depth	The time and date at which the queue reached its minimum or maximum depth.
Type	The direction of message flow: inbound or outbound.
Total Messages	The total number of messages in the queue at one point since the last server reboot.
Bytes Received	The total number of bytes processed by the queue since the last server reboot.
Last Activity Time	The time at which the most recent message was added to the queue since the last server reboot.

Data Change Notification Statistics

Data change notification (DCN) statistics monitor notifications that are received by Unwired Server from the enterprise information server. Specifically, DCN monitoring reports which packages and sync groups are affected by notifications, and how quickly these are processed by the server.

Monitoring DCN statistics allows you to troubleshoot and diagnose performance issues if, for example, the cache is not being updated quickly enough. These statistics help to identify which packages took longest to process data changes, as well as times of peak performance or strain on the system.

Data Change Notification History Statistics

Historical information for data change notifications (DCNs) consists of past notification details for monitored packages. Detailed data provides specific information on past notification activity for packages, and identifies which server data was affected.

Details about past notification events are organized into these categories:

Category	Description
Domain	The domain to which the package affected by the DCN belongs.
Package	The name of the package containing data changes.

Category	Description
MBO	The name of the MBO to which the notification applied.
Notification Time	The date and time that Unwired Server received the DCN.
Processing Time	The time that Unwired Server used to process the DCN.

Data Change Notification Performance Statistics

Data change notification (DCN) performance statistics consist of key performance indicators that reflect the efficiency of notification processing by Unwired Server.

Performance monitoring highlights key totals and identifies average, minimum, and maximum values for primary activities. These calculations are dynamic, and are based on the data currently available in monitoring database for the specified time period.

All values in this table (totals, averages, maximums, minimums) apply to the specific time period you indicate:

Key performance indicator	Description
Total Notifications	The total number of notifications sent by the enterprise information system to Unwired Server.
Average/Minimum/Maximum Processing Time	The average, minimum, or maximum amount of time Unwired Server took to process a DCN.
Time at Minimum/Maximum Message Processing Time	The time of day at which the shortest or longest DCN processing event completed.
Time of Last Notification Received	The time at which the most recent DCN was received by Unwired Server.
MBO with Minimum/Maximum Notification Processing Time	The name of the package and associated mobile business object (MBO) with the shortest or longest notification processing time.

Device Notification Statistics

Device notification statistics provide data about the occurrence and frequency of notifications sent from Unwired Server to replication-based synchronization (RBS) devices. Historical device notification monitoring reports on the packages, synchronization groups, and devices affected by RBS synchronization requests in a given time frame. Performance-related device notification data provides a general indication of the efficiency of notification processing and the total demand of synchronization requests on the system.

Device Notification History Statistics

Historical information for device notifications provides specific information on past device notifications, indicating which packages, synchronization groups, and devices were involved in synchronization requests.

Details about past device notification events fall into these categories:

Category	Description
Domain	The domain to which the package affected by the device notification belongs.
Package	The name of the package containing data changes.
Synchronization group	The synchronization group that the package belongs to.
Device ID	The ID number of the mobile device participating in the synchronization request.
Generation time	The date and time that Unwired Server generated the device notification.
User	The name of the user associated with the device ID.

Device Notification Performance Statistics

Device notification performance statistics provide a general indication of the efficiency of notification processing and the total demand of synchronization requests on the system.

Performance monitoring highlights key totals and identifies average, minimum, and maximum values for primary activities. These calculations are dynamic, and are based on the data currently available in monitoring database for the specified time period.

All values in this table (totals, averages, maximums, minimums) apply to the specific time period you indicate:

KPI	Description
Synchronization Group for Maximum Notifications	The synchronization group for which the maximum number of notifications were sent.
Package for Maximum Notifications	The package for which the greatest number of device notifications were sent.
Total Notifications	The total number of device notifications sent from Unwired Server to devices.
Total Distinct Users	The total number of users that received device notifications. This value comprises only individual users, and does not count multiple synchronizations requested by the same user.

KPI	Description
Total Distinct Devices	The total number of devices that received device notifications. This is distinct from Total Distinct Users, because a single user name can be associated with multiple devices.
Enabled Subscriptions	The total number of replication subscriptions for which notifications are generated.
Time at Last Notification	The time at which the last device notification was sent by Unwired Server.
Outstanding Subscriptions	The total number of replication subscriptions, both enabled and disabled.

Package Statistics

Package statistics reflect response times for replication-based and messaging-based synchronization packages.

This type of monitoring uses key performance indicators to provide data on the efficiency of response by Unwired Server to synchronization requests. These calculations are dynamic, and are based on the data currently available in monitoring database for the specified time period.

Replication Package Statistics

Replication package statistics consist of key performance indicators (KPIs) that reflect the overall function of the application environment at the cluster or domain level. The statistics highlight key totals and identify average, minimum, and maximum values for primary activities.

These key indicators monitor replication packages:

Note: These KPIs are not applicable at the MBO level.

- Total Bytes Received
- Total Bytes Sent
- Total Operation Replays

KPI	Description
Total Devices	The total number of devices involved in synchronization. This total includes the same user multiple times if he or she has multiple devices. The value comprises individual devices, and does not count multiple synchronizations requested by the same device.
Total Rows Sent	The total number of rows sent during package synchronization.

KPI	Description
Total Rows Received	The total number of rows received during package synchronization.
Total Errors	The total number of errors that took place across all synchronizations.
Total Bytes Received	The total number of bytes uploaded from clients to Unwired Server.
Total Bytes Sent	The total number of bytes downloaded by clients from Unwired Server.
Average/Minimum/Maximum Synchronization Time	The average, minimum, or maximum amount of time Unwired Server took to finish a complete synchronization.
Time at Minimum/Maximum Synchronization Time	The time at which the shortest or longest synchronization completed.
Total Synchronization Requests	The total number of sync requests initiated by a client.
Total Operation Replays	The total number of operation replays performed by clients on MBOs.

Messaging Package Statistics

Messaging package statistics consist of key performance indicators (KPIs) that reflect the overall function of the application environment at the cluster or domain level. The statistics highlight key totals and identify average, minimum, and maximum values for primary activities.

Note: These KPIs are not applicable at the MBO level:

- Total Subscription Commands
- Total Devices

These key indicators monitor messaging packages:

KPI	Description
Total Subscription Commands	The total number of subscription commands sent from clients to the server.
Total Devices	The total number of devices involved in synchronization. This total includes the same user multiple times if he or she has multiple devices. The value comprises individual devices, and does not count multiple synchronizations requested by the same device.

KPI	Description
Average/Minimum/Maximum Message Processing Time	The average, minimum, or maximum amount of time Unwired Server took to respond to a synchronization request message.
Time at Minimum/Maximum Processing Time	The time at which the shortest or longest response time completed.
Total Inbound Messages	The total number of messages sent from clients to Unwired Server.
Total Outbound Messages	The total number of messages sent from Unwired Server to clients.
Total Operation Replays	The total number of operation replays performed by clients on mobile business objects (MBOs).
Total Errors	The total number of errors that took place across all synchronizations.
Total Data Push	The total amount of data transmitted from the server to clients.

User Statistics

User statistics consist of key performance indicators that reflect the overall activity of application users.

User statistics can be filtered to include users who belong to a particular security configuration. This type of monitoring highlights key totals and identifies average, minimum, and maximum values for primary user activities. These calculations are dynamic, and are based on the data currently available in monitoring database for the specified time period.

Note: These statistics are not supported for Sybase Mobile CRM and Sybase Mobile Workflow for SAP application users.

Replication User Statistics

Replication user statistics reflect the synchronization activity of a group of replication-based synchronization users belonging to a specified security configuration. These statistics include general activity-related information on a per-user basis.

These key indicators monitor replication users:

KPI	Description
Total Synchronization Requests	The total number of sync requests initiated by a client.
Total Rows Received	The total number of rows received during package synchronization.
Total Rows Sent	The total number of rows sent during package synchronization.
Total Bytes Received	The total number of bytes uploaded from clients to Unwired Server.

KPI	Description
Total Bytes Sent	The total number of bytes downloaded by clients from Unwired Server.
Average/Minimum/Maximum Synchronization Time	The average, minimum, or maximum amount of time Unwired Server took to complete a synchronization request.
Time at Maximum/Minimum Synchronization Time	The time at which the fastest or slowest synchronization is completed.
Total Operation Replays	The total number of operation replays performed by user of mobile business objects (MBOs).
Total Errors	The total number of errors that took place across all synchronizations.
Total Devices	The total number of devices involved in synchronization. This total includes the same user multiple times if he or she has multiple devices. The value comprises individual devices, and does not count multiple synchronizations requested by the same device.

Messaging User Statistics

Messaging user statistics reflect the synchronization activity of a group of messaging-based synchronization users belonging to a specified security configuration. These statistics include general activity-related information on a per-user basis.

These key indicators monitor messaging users:

KPI	Description
Total Devices	The total number of devices involved in synchronization. This total includes the same user multiple times if he or she has multiple devices. The value comprises individual devices, and does not count multiple synchronizations requested by the same device.
Average/Minimum/Maximum Message Processing Time	The average, minimum, or maximum amount of time Unwired Server took to respond to a sync request message.
Time at Minimum/Maximum Message Processing Time	The time of day at which the shortest or longest message processing event completed.
Total Inbound Messages	The total number of messages sent from clients to Unwired Server.
Total Outbound Messages	The total number of messages sent from Unwired Server to clients.
Total Operation Replays	The total number of operation replays performed by clients on mobile business objects (MBOs).

KPI	Description
Total Errors	The total number of errors that took place across all synchronizations.
Total Subscription Commands	The total number of subscription commands sent from clients to the server.
Total Data Push	The total number of import data messages.

Security Log Statistics

The security log reflects the authentication history of users either across the cluster, or filtered according to domain, during a specified time period. These statistics allow you to diagnose and troubleshoot connection or authentication problems on a per-user basis.

User security data falls into these categories:

Category	Description
User	The user name
Security Configuration	The security configuration to which the device user belongs
Time	The time at which the authentication request took place
Result	The outcome of the authentication request: success or failure
Device ID	The device ID associated with the user
Package	The package the user was attempting to access
Domain	The domain the user was attempting to access

Cache Statistics

Cache statistics provide a granular view of cache activity eithr at the domain or package level, particularly in the areas of cache performance, mobile business object (MBO) status, and cache group status.

Cache statistics report on performance at the domain, package, MBO, and cache group levels to allow administrators to obtain different information according to the level of specificity required. These calculations are dynamic, and are based on the data currently available in monitoring database for the specified time period.

Note: These statistics are not supported for Sybase Mobile CRM and Sybase Mobile Workflow for SAP application users.

MBO Statistics

Mobile business object (MBO) status monitoring reports on cache activity at the MBO level, and thus, reflects activity for single mobile business objects.

Select **Package level MBO** to view the following key performance indicators:

Key performance indicator	Description
Cache Group	The name of the group of MBOs associated with this cache activity.
MBO	The name of the single mobile business object associated with this cache activity.
Number of Rows	The number of rows affected by the cache refresh.
Cache Hits	The number of scheduled cache queries that occurred in the supplied date range.
Cache Misses	The number of on-demand cache or cache partition refreshes that occurred in the supplied date range.
Access Count	The number of cache queries that occurred in the supplied date range.
Minimum/Maximum/Average Wait Time	The minimum, maximum, or average duration of cache queries in the supplied date range. This time does not include the time required to refresh the cache in a cache "miss" scenario. Instead Minimum/Maximum/Average Full Refresh Time exposes this data.
Minimum/Maximum/Average Full Refresh Time	The minimum, maximum, or average duration of on-demand and scheduled full refresh activities in the supplied date range.

Cache Group Status Statistics

Cache group status statistics provide monitoring data about cache activity at the cache group level. The data reflects activity for all mobile business objects (MBOs) belonging to a cache group.

Select **Package level cache group** to view the following key performance indicators (KPIs):

KPI	Description
Package	The name of the package to which the associated cache group belongs
Cache Group	The name of the group of MBOs associated with the cache activity
Number of Rows	The number of rows in the cache table of the MBO
Last Full Refresh Time	The last time the cache or cache partition was fully refreshed

KPI	Description
Last Update Time	The last time a row in the cache was updated for any reason (row-level refresh, full refresh, partitioned refresh, alternate read, or data change notification)
Last Invalidate Time	The last time the cache was invalidated
Cache Coherency Window	The data validity time period for the cache group, in seconds. Can span any value in this range: • 0 shows that data is always retrieved on-demand for each client. • 2049840000 shows that the cache never expires. This occurs when you set the on-demand cache group to NEVER expire or scheduled cache group to NEVER repeat.

Refining Scope with Filters, Sorting, and Views

You can refine any view in the Sybase Contorl Center monitoring to show selected details of particular relevance.

Use these to narrow the scope of data represented in the object tabs.

1. To sort table data alphanumerically by column, click the column title. Sorting is available on all tabs of the monitoring node.
2. You can filter data by either:
   - Time – set the start and end day and time for which to show data, or,
   - Domain – check **Show Current Filter** to set the domain for which to show data (as opposed to showing all domains, which is the default). You can optionally sort these results by package name, synchronization phase, operation, and so on by selecting the corresponding option from the **Sort By** box.
3. For historical data on application tabs, you can also toggle between detail or summary views by selecting the option of the same name. Detail views show specific details of each application and each operation (update, download), whereas summaries include aggregates of each application (total messages sent, total bytes received).

Exporting Monitoring Data

Save a segment of monitoring data to a location outside of the monitoring database. Export data to back up information, particularly before purging it from the database, or to perform closer analysis of the data in a spreadsheet application.

This option is especially useful when you need to share monitoring data with other administrators and tenants. Since this task can be time-consuming, depending upon the size of the data being exported, Sybase recommends that you export the data in segments or perform the export at a time when Sybase Control Center is not in use.

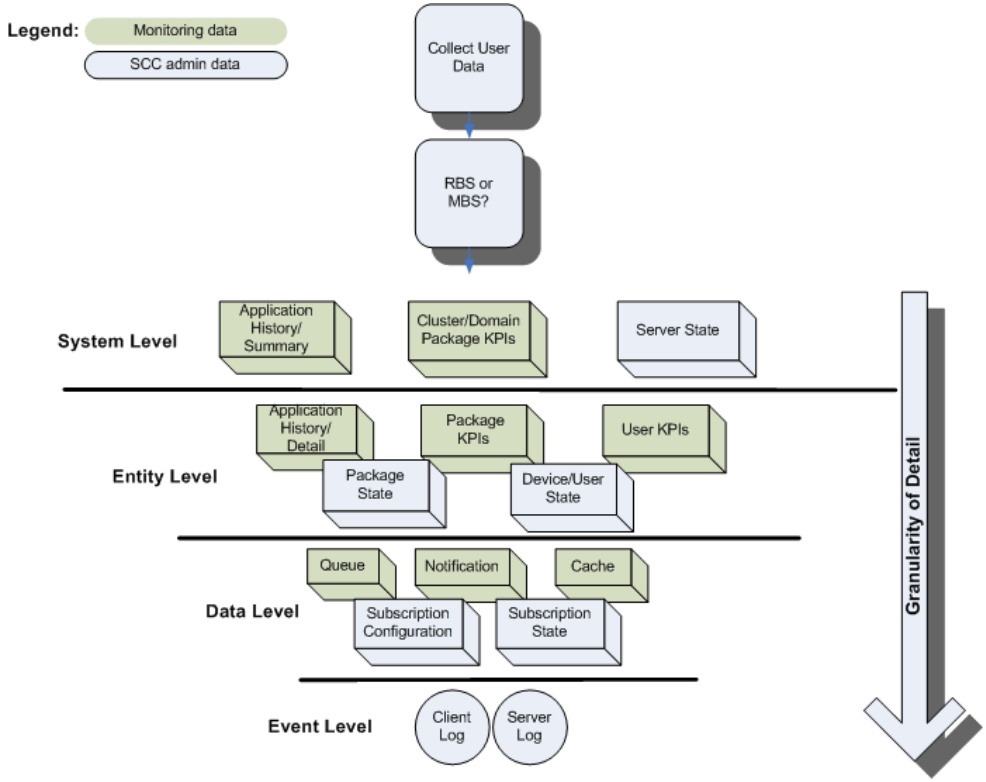
1. In the left navigation pane, select **Monitoring**.
2. In the right administration pane, select the tab corresponding to the monitoring data you want to view.
3. Perform a search using the appropriate criteria to obtain the desired monitoring data.
4. Click **Export**.
5. Select a file type for the exported data (CSV or XML), and click **Next**.
6. Click **Finish**.
7. In the file browser dialog, select a save location and enter a unique file name.
8. Click **OK**.
   All monitoring data retrieved by the search is saved to the file you specify in step 7.

System Diagnostics

Diagnosing performance issues or troubleshooting errors involves reviewing system-wide data, and typically begins with information captured by the Monitoring node in the Sybase Control Center Unwired Platform administration perspective. However, it can extend to any and all state and log data that is available to the administrator.

There is no rigidly defined troubleshooting path for administrators to take. Instead, data is reviewed and analyzed component by component until a comprehensive picture of the system emerges and gives the administrator enough symptoms to diagnose the problem.

As shown by the illustration below, monitoring and state data is collected on the platform at various levels to create an inverted pyramid of potential diagnositc data. Using this pyramid, an administrator can scale up and scale down in terms of the specificity of detail, depending on the issue being investigated. Subsequent scenarios in this section use this illustration to show you which diagnostic components may help you diagnose issues or errors.



See also

- *Device Application Performance or Issue Analysis* on page 239
- *Access Denied Analysis* on page 243
- *Data Update Failure Analysis* on page 244

Collecting Data

When diagnosing application errors and issues, the user need to provide some intial troubleshooting data that helps to identify which records and events captured by the Sybase Control Center monitoring tool should be more carefully analyzed for telling symptoms.

1. Contact the user or use an issue reporting tool that collects:
   - application type (replication or messaging)
   - and packages that make up the application
   - the user ID
   - the device ID
   - the time of the error/issue/period of interest (roughly)

2. Use the package type to start the analysis of events, as well as the package name. Other information will be necessary the more detailed your investigation becomes.

Device Application Performance or Issue Analysis

If a device user reports lagging performance or errors in a deployed application, there are a series of diagnostics the administrator can perform to investigate the problem.

Symptoms that are not revealed by the monitoring tables are likely to require a review of administrative data from other parts of Unwired Platform.

Check for	Using	See
Synchronization performance issues	Package historical data to see if high volumes or frequent operations are causing the issue or problem	*Checking Basic Application Statistics*
	Statistical information collected for the package	*Checking Package Statistics Overall*
	User statistics to see what else the user is doing at the time	*Checking User KPIs for Other Data Usage*
	Sybase Control Center server administration data to see the responsiveness of the server	*Checking Server Responsiveness in Sybase Control Center*
Unwired Server errors or failures	System logs to see if there are messages indicating whether something has failed	*Looking for Errors and Failures in SCC*

See also

- *Collecting Data* on page 238
- *Checking Package/User Histories* on page 239
- *Checking Overall Package Statistics* on page 240
- *Checking User KPIs for Other Data Usage* on page 241
- *Checking Server Responsiveness in Sybase Control Center* on page 242
- *Looking for Errors and Failures in SCC* on page 242

Checking Package/User Histories

Always begin analyzing application errors or issues by checking high-level application statistics. The goal is to see if there are any obvious outliers in typical or expected application performance. The more familiar an administrator is with the environment, the more easily the administrator can identify abnormal or unexpected behavior or performance.

If time has elapsed since an issue was reported, start by checking historical information for the entire package. Drill down into suspect rows.

1. In Sybase Control Center, click the Monitoring node, then click the tab that corresponds to the application type you are investigating, either **Replication** or **Messaging**.
2. Click **History**, then choose **Summary**, to display an aggregated history for the package.
3. Select **Show current filter**, to narrow the results. Using the troubleshooting data you gathered from the user, in the filter pane:
   a) Select the domain to which the package is deployed.
   b) Select the package name from the list of packages deployed to the domain.

   Note: In the **Domain** and **Packages** fields, you can start to type a package or domain name to narrow the list to names beginning with the characters you enter.
4. Click the **User** column to sort in alphabetical (ascending or descending) order.
5. Refine entries to those that fall within the documented time frame. Set the **Start Date**, **Start Time** and **Finish Date**, **Finish Time** to restrict the data to the required duration.
6. Determine whether the volumes are high for any of the cells in a row. If so, further investigate the details of that package/user pairing row.
7. On the History tab, choose **Detail view** and locate the package/user row you are investigating. This view details the synchronization request, so you can find out where in the transaction chain the problem is arising. It helps you to identify what happened at the entity level.
8. Look at these columns:

   - Total Rows Sent to see if the number of data rows being synchronized is high or low.
   - Payload size to see the number of bytes downloaded to the device for each event.

   These counters may indicate that there is a lot of data being transferred during synchronization and may require some intervention to improve performance. You can then look at the entity or phase where the problem is occurring, and whether or not there is a delay of any kind that might indicate the nature of the problem. For example, perhaps large volumes of uploaded data may be delaying the download phase. However, it may also be that this behavior is normal, and the performance lag transitory.

Next

If nothing of interest is revealed, continue by checking the package statistics for all users.

Checking Overall Package Statistics

If the package history does not reveal a high amount of volume (data or frequency of operations), proceed with evaluating the package statistics for the entire environment. Package statistics can reveal issues caused by the number of device users in the environment, and how the package is performing in environment in general.

1. In Sybase Control Center, click the Monitoring node, then click the **Package Statistics** tab.
2. Choose the type of package you want to retrieve KPIs for: messaging or replication.
3. Set the **Start Date**, **Start Time** and **Finish Date**, **Finish Time** so KPIs are aggregated during the specified time frame.
4. In the navigation tree, expand clusters, domains, and servers, until you locate the package to investigate.
5. Click the package name and review:

   - Total Data Push
   - Time at Minimum/Maximum/Average Synchronization
   - Total Devices

   Compare the findings to see whether or not these results are revealing. Also check the number of concurrent users so you can gauge the full range of activity (for example, use the navigation tree to scope data push results to the domain and cluster): if multiple device users are using similar packages to synchronize a similar set of data, then your system could be experiencing lags due to high demands on the server cache.

Next

If package statistics do not provide useful information about data demands, evaluate the data demands of individual users.

Checking User KPIs for Other Data Usage

If the application is not downloading large amounts of data, further investigate user-based key performance indicators (KPIs) to see if the performance issue is related to the user who is performing some other data-related action that may not be related to the package for which the issue was reported.

1. In Sybase Control Center, click the Monitoring node, then click the **User Statistics** tab.
2. Choose the type of package for which to retrieve KPIs: messaging or replication.
3. Set the **Start Date**, **Start Time** and **Finish Date**, **Finish Time** so KPIs are aggregated for the specified time frame.
4. If the package is deployed to a particular domain, choose the domain name you require.
5. Select the user who reported the issue.
6. Review these values :

   - Total Data Push
   - Time at Minimum/Maximum/Average Synchronization Time

   A high data push value might indicate that at some point, data demands were large. If the average synchronization and minimum synchronization times fall within normal ranges but the maximum is high, it may indicate that data volumes have created a synchronization performance lag for the package. Low data delivery for the package (determined during

package-level analysis), but high values for the individual user may suggest an unusual demand pattern which may be harmless or require further investigation. Simultaneous synchronization demands may be impacting the performance of all applications on the device.

Next

If user data demands reveal no abnormalities, continue by checking administration data in other areas of Sybase Control Center, such as Device User administrative data and subscription configuration.

Checking Server Responsiveness in Sybase Control Center

If information concerning the package, users, and the environment seem to all return normal results, you may want to investigate Unwired Server data, by checking administration data in other parts of Sybase Control Center (SCC).

1. In Sybase Control Center, click the Device Users node, then choose the type of applications that are used on the device:
   - **Unified** for both application types
   - **RBS** for replication applications
   - **MBS** for messaging applications
2. Sort the results by ascending or descending device ID order by clicking the **Device ID** column.
3. Verify whether data is being delivered by looking in the **Pending Items** or **Last Delivery** columns.
4. Expand the **Packages** node, select the package name, then choose the **Subscriptions** tab to compare the delivery results with the subscription status. Use the **Last Server Response** column to see when the last message was sent from the server to the device.

Next

If the message did not transmit and the server appears responsive, continue evaluation by *Looking for Errors and Failures in SCC*.

Looking for Errors and Failures in SCC

If data in the Monitoring node reveals nothing unusual, extend your analysis to other nodes in Sybase Control Center (SCC). Look for explicit errors or failures that may be recorded elsewhere.

This will help you determine where the error may exist: the device, the Unwired Server, or the enterprise information system (EIS) data source.

1. In Sybase Control Center, click the **Packages** node.
2. To check whether the problem exists in either the device or the EIS data source:

a) Click the **Client Log** tab.
b) Check the Operation and Message columns and look for any operation replays that failed with error code 500, and the reason for the failure. For example, the client log shows all logs on device. For failed operations on the device, check the details of error message and assess what kind of error may have occurred.

3. To check whether the problem exists in either the server or the EIS data source:
   a) Expand the package tree until MBOs and operations for the package complete the tree navigation for the package.
   b) For each MBO and operation, select the node in the navigation tree, then click the **History** tab.
   c) Set the **Start Date**, **Start Time** and **Finish Date**, **Finish Time** to restrict the data to the duration you require.
   d) Review the data and look for any errors and the potential causes of those errors. You can check the error history, the exception or error about the operation is listed in the details of that error. Use the message to reveal the issue and coordinate with the development team as required.

Access Denied Analysis

If a user reports an `access is denied` error, the administrator can check the security log, and from there, validate the package's security configuration.

Checking the Security Log

Validate `access is denied` messages by checking the security log.

1. In Sybase Control Center, click the Monitoring node, then click the **User Statistics** tab.
2. Click **Security Log**.
3. Set the **Start Date**, **Start Time** and **Finish Date**, **Finish Time** to restrict the data to the specified time frame.
4. Click the **Result** column to sort rows by result type.
5. Locate any authentication failures or access denied events that are logged for the user who reported the error.
6. If you find any errors in the result column, check package names and security configurations. Then check to see if a similar result is reported for other user names. Also verify if the error persists; a heavily loaded service could cause a transient error. Transient errors are generally resolved retrying the connection.

Next

If there are no errors, investigate the security setup for the pair.

Validating Security Setup

If users are reporting `access is denied` errors where access should be allowed, validate your security setup. A security configuration is defined at the cluster level by the platform

administrator, then assigned at domain and package levels by either administrator type, so it may take some analysis to determine where the problem is occurring.

Use the security log to evaluate the properties of the assigned security configuration.

1. In Sybase Control Center, expand the navigation tree in the Unwired Platform administration perspective until you locate the security configuration that generated the error. It appears either in the **Domains > <domain_name> > Security** folder or the **Security** folder at the cluster root.
2. Select the configuration you are investigating.
   - If the security configuration is assigned to a domain, validate that the role mapping is correct:
     - If the Unwired Platform user is the exact name of the user in the security repository, then no mapping is required.
     - If the Unwired Platform user differs, even slightly, then logical roles used by the package, and physical roles used in the repository must be manually mapped.
   - Review the existing security policy with the security administrator to ensure that privileges are set correctly.

Data Update Failure Analysis

If a user reports an unreceived push notification (for replication packages only), or that data appears to be updating incorrectly, administrators should follow a similar process as documented for the device application performance analysis scenario.

The adminstrator needs to first assess the synchronization performance by checking data as documented in *Device Application Performance Analysis*. From there you can specifically zero in on the device push notifications and cache statistics to see if there's an issue with how data updates are delivered (as device notifications) and configured (as subscriptions).

Checking Last Notification Timestamp

If a user reports that data is not current, you can assume that either replication-based synchronization or synchronization device notifications are not working as designed. To help you ascertain which, start by checking notifications.

Prerequisites

Before investigating notification or subscription issues, confirm that synchronization is behaving normally.

Task

1. In Sybase Control Center, click the Monitoring node, then click the **Device Notifications** tab.
2. Select **History**.
3. Click **User** to sort on user name in ascending or descending alphabetical order.

4. Set the **Start Date**, **Start Time** and **Finish Date**, **Finish Time** to restrict the data to the time frame you specify.
5. In the Time of Notification column, ensure that the timestamp was recently recorded, then compare the start and end time of the last known synchronization. Device notification should always occur before a synchronization. If a notification is not received by a user, they may assume that data is not updating correctly.

Checking Cache Statistics

Cache statistics provide a granular view of cache activity, particularly in the areas of cache performance, mobile business object (MBO) status, and cache group status at different levels of use in the mobility environment.

1. In Sybase Control Center, click the Monitoring node, then click the **User Statistics** tab.
2. Set the **Start Date**, **Start Time** and **Finish Date**, **Finish Time** so KPIs are aggregated during the specified time frame.
3. Choose the level of cache activity to investigate. Sybase recommends that you:
   - Select **Package level cache group** to investigate cache activity for the cache group, especially if the cache group is used by multiple MBOs. Review the last refresh time to see if data has recently been refreshed from the enterprise information system (EIS), and also check to see that the affected rows are reasonable for the package.
   - Select **Package level MBO** to investigate cache activity at the MBO level. Review data related to cached rows for the MBO, including the number of cache hits and misses. The number of hits should typically be higher than the number of misses. If you do not see any hits, or see a lower number of hits than misses, the notification schedule may not working as designed. See *Validating States and Settings* to see how the subscription that schedules push notifications is set up.

Validating States and Settings of Features that Update Data in SCC

If synchronization occurs after a notification, or if a notification arrives but data is not updated, both symptoms require you to evaluate the subscription settings.

Most importantly, evaluate how the cache group interval, sync group interval, and notification threshold properties are configured. If one of these items is mistimed, the user is likely to experience an unlikely result.

1. Check the state and settings of the cache group used by the package.
   a) In Sybase Control Center, expand the Packages node, select the package name, then choose the **Cache Group** tab.
   b) Select the box beside the cache group name and click **Properties**.
   c) Verify:

    - That the group state has a valid status. Cache status can be suspended, available, or refreshing. Set cache group so that it is better coordinated with the change detection interval.
    - The cache interval or schedule used to refresh data in the Unwired Server cache based on changes in the back-end EIS data source. Make note of that refresh interval to use in the next step.
2. Select the **Subscriptions** tab and verify:
    - That the user subscription has not been removed or suspended.
    - For replication-based synchronization, that push synchronization is used as required. Follow up with the user to ensure that push synchronization is enabled on the device.
    - That the synchronization group interval is configured appropriately based on the cache interval or schedule repeat. This value determines how frequently change detection occurs.
    - For replication-based synchronization, that the notification threshold is configured appropriately based on the synchronization group interval. This value determines how frequently a user is notified of updated server cache data.

System Logs

Unwired Platform uses multiple logs to record events that are useful for administrators who are monitoring the environment, and maintaining the health of components.

Administrators should regularly review log messages that can be recorded at differing levels of severity.

Messages in various platform logs can provide information about:

- Configuration changes
- Successful and unsuccessful system operations (for example, deployment, synchronization and so on)
- System events and failures

Log File Locations

Use a text editor to review log files from the command line using a text editor if Sybase Control Center is not available, or to concentrate your review to a particular log.

Table 21. Log file locations

Log type	Location
Unwired Server	Aggregated Unwired Server logs, including replication-based synchronization events and SYSAM issues: `<UnwiredPlatform_InstallDir>\UnwiredPlatform\Servers\UnwiredServer\logs` Messaging service log details: `<UnwiredPlatform_InstallDir>\UnwiredPlatform\Servers\MessagingServer\Data\Log` `<UnwiredPlatform_InstallDir>\UnwiredPlatform\Servers\MessagingServer\Trace` Mobile Workflow tracing logs: `<UnwiredPlatform_InstallDir>\UnwiredPlatform\Servers\UnwiredServer\logs\WorkflowClient`
Sybase Control Center for Sybase Unwired Platform	Sybase Control Center agent log: `<UnwiredPlatform_InstallDir>\SCC-XX\log\agent.log` Repository log: `<UnwiredPlatform_InstallDir>\SCC-XX\log\repository.log` Request logs: `<UnwiredPlatform_InstallDir>\SCC-XX\services\EmbeddedWebContainer\log\request-<yyyy_mm_dd>.log` Database message: `<UnwiredPlatform_InstallDir>\SCC-XX\services\Repository\scc_repository.slg`

Log type	Location

Relay server and RSOE	Default log details:
	`%temp%\ias_relay_server_host.log`
	`%temp%` is the Windows environment variable, for example, `C:\WINDOWS\Temp`.
	Note: In the case of IIS Web servers, the log file is `C:\WINDOWS\system32\LogFiles`. However, this location can be configurable in IIS Manager.
	RSOE log files, by default:
	`<UnwiredPlatform_InstallDir>\UnwiredPlatform\Servers\UnwiredServer\logs`
	These log files are generated in the `\logs` directory for the corresponding RSOE:
	• `rsoe.log`
	• `msgrsoe.log`
	• `webserver_rsoe.log`

Installer	`<UnwiredPlatform_InstallDir>`
	`<UnwiredPlatform_InstallDir>\UnwiredPlatform\InstallLogs`
	`<UnwiredPlatform_InstallDir>\UnwiredPlatform\InstallLogs\silentInstall`

Domain	Added to the monitoring database; viewable in Sybase Control Center
Consolidated database	By default, consolidated database errors are logged in the `errorlog.txt` file, located in `<UnwiredPlatform_InstallDir>\UnwiredPlatform\Servers\UnwiredServer`.

Message Syntax

Unwired Platform log messages typically consist of a standard set of message elements.

- Date (year-month-day when the event occurred)
- Time (hour:minute:second when the event occurred)
- Component/module (where the event occurred)

- Severity level
- Message type (a code number associated with the severity level)
- Message text (content of the event message)

Messages may also include the administrator's login name, if the administrator performed an action.

Severity Levels and Descriptions

Unwired Platform can record messages at various severity levels.

You can control the level of messages that get recorded for Unwired Server from Sybase Control Center. See *Sybase Control Center online help > Configure > Configuring Unwired Platform > Unwired Server > Server Logs Configuring Server Log Settings*.

Log messages that typically get recorded include:

Level	Description
Debug	Detailed information useful for debugging purposes.
Info	General system information.
Warn	Messages about conditions that might affect the functionality of the component, such as a failure to connect to servers or authentication failures, timeouts, and successes.
Error	Messages about conditions that may require immediate attention.

Enabling and Configuring Logging

If you are having a problem with either the Unwired Platform runtime or Sybase Control Center administration, you might be able to discover the cause of the problem by enabling or changing the logging level so that more events are recorded.

Configuring Server Log Settings

Configure server log properties to specify the amount of detail that is written to the log, as well as the duration of the server log life cycle.

How changes are applied in a cluster depends on whether you are configuring a primary or secondary server. Sybase recommends you only configure log settings on the primary server. If you change the setting on a secondary server, the configuration is updated only for that server and is temporary (eventually the primary settings are propagated to all servers in the cluster).

Additionally, you should always use Sybase Control Center to configure server logs. If you manually edit the configuration file, especially on secondary servers in a cluster, the servers may not restart correctly once shut down.

1. In the left navigation pane, expand the **Servers** folder and select the server to configure.
2. Select **Log**.
3. In the right administration pane, click the **Settings** tab.
4. Set the server log size and backup behavior that jointly determine the server log life cycle.
   a) Set the **Maximum file size**, in kilobytes, megabytes, or gigabytes, to specify the maximum size that a file can reach before a new one is created. The default is 10MB.

      Alternatively, select **No limit** to log all events in the same file, with no maximum size.
   b) Set the **Maximum backup index** to determine how many log files are backed up before the oldest file is deleted. The index number you choose must be a positive integer between 1 and 65535. The default is 10 files.

      Alternatively, select **No limit** to retain all log files.
5. For each of the listed components, choose one of these log levels:

Log level	Messages logged
All	Complete system information
Trace	Finer-grained informational events than debug
Debug	Very fine-grained system information, warnings, and all errors
Info	General system information, warnings, and all errors
Warn	Warnings and all errors
Error	Errors only
Console	Messages that appear in the administration console only (when Unwired Server is running in non-service mode)
Off	Do not record any messages

   The default log levels are:

Component	Default Log Level
MMS	Info
MSG	Info
Security	Info
Mobilink	Info
DataServices	Info
Other	Warn

6. Click **Save**.

Log messages are recorded as specified by the settings you choose. The log file is located in: `<UnwiredPlatform_InstallDir>\<UnwiredPlatform>\Servers\UnwiredServer\logs\<hostname>-server.log`.

Log life cycle default example

If you keep the default maximum file size and default index, an Unwired Server writes to the log file until 10MB of data has been recorded. As soon as the file exceeds this value, a new version of the log file is created (for example, the first one is `<hostname>-server.log.1`). The contents of the original log are backed up into this new file. When the `<hostname>-server.log` file again reaches its limit:

1. The contents of `<hostname>-server.log.1` are copied to `<hostname>-server.log.2`.
2. The contents of `<hostname>-server.log` are copied to `<hostname>-server.log.1`.
3. A new copy of `<hostname>-server.log` is created.

This rollover pattern continues until the backup index value is reached, with the oldest log being deleted. If the backup index is 10, then `<hostname>-server.log.10` is the file removed, and all other logs roll up to create room for the new file.

See also

- *Enabling and Configuring Domain Logging* on page 252
- *Configuring Sybase Control Center Logging for Performance Diagnostics* on page 253
- *Configuring Messaging and Mobile Workflow Runtime Logging* on page 255
- *Configuring Messaging Device Logging* on page 256
- *Configuring RSOE Logging* on page 258
- *Configuring and Enabling Relay Server Logging* on page 258
- *Enabling Custom Log4j Logging* on page 259

Enabling and Disabling HTTP Request Logging for DCNs

Configure HTTP logging to record request event information logged by data change notifiications (DCNs). By default, HTTP logging for DCNs is enabled, and prints output to `<UnwiredPlatform_InstallDir>/Servers/UnwiredServer/logs/<server.name>-http.log`.

You can disable HTTP logs if you do not use DCNs.

1. Open `<UnwiredPlatform_InstallDir>/Servers/UnwiredServer/Repository/Instance/com/sybase/djc/server/ApplicationServer/${yourserver}.properties`.
2. Delete the `enableHttpRequestLog` line.

3. Save the file.
4. Restart Unwired Server.

Increasing the Maximum Post Size

Increase the size of an HTTP request to handle larger data change notifications.

The default size of an HTTP request is 10 MB. The default size is set in the `jetty-web.xml` file.

1. Stop all Sybase Unwired Platform services.
2. Open the `jetty-web.xml` file, which is located in `<UnwiredPlatform_InstallDir>\Unwired Platform\Servers\UnwiredServer\deploy\webapps\dcn\WEB-INF\`.
3. Find the property, **<Set name="maxFormContentSize" type="int">10000000</Set>** and increase the value up to 100MB, for example: **<Set name="maxFormContentSize" type="int">100000000</Set>**.
4. Save the file.
5. Restart Sybase Unwired Platform services.

Enabling and Configuring Domain Logging

Activate or deactivate domain logging in Sybase Control Center, and configure domain log autopurge settings for all nodes in a cluster. Domain logging collects data that pertains to the activities of all packages in a domain. You must have administrator privileges to configure domain logging.

First, domain-level logging must be enabled by a platform administrator. Domain-level logging controls whether package-level logging captures data. Then either the platform administrator or the domain administrator can enable logging on a per-package basis from the Packages node of Sybase Control Center. See *Sybase Control Center online help > Configure > Configuring Unwired Platform > Packages > Enabling Package Logging*.

1. Open Sybase Control Center.
2. In the left navigation pane, expand the **Domains** folder and select the domain for which to configure log settings.
3. Select **Log**.
4. In the right administration pane, select the **Settings** tab.
5. Select one of:
   - **Enable** – activate domain logging in Sybase Control Center.
   - **Disable** – turn off domain logging.
6. Set the autopurge threshold by entering the length of time (in days) to retain domain log data.
7. Click **Save**.

See also

- *Configuring Server Log Settings* on page 249
- *Configuring Sybase Control Center Logging for Performance Diagnostics* on page 253
- *Configuring Messaging and Mobile Workflow Runtime Logging* on page 255
- *Configuring Messaging Device Logging* on page 256
- *Configuring RSOE Logging* on page 258
- *Configuring and Enabling Relay Server Logging* on page 258
- *Enabling Custom Log4j Logging* on page 259

Exporting Domain Log Data

Save a segment of domain log data to a location outside of the monitoring database. Export data to back up information or to perform closer analysis of the data in a spreadsheet application.

1. Open Sybase Control Center.
2. In the left navigation pane, expand the **Domains** folder and select the domain to configure.
3. Select **Log**.
4. Perform a search to obtain the desired log data.
5. Click **Export**.
6. Select a file type for the exported data (.TXT, or .XML) and click **Next**.
7. Click **Finish**.
8. In the file browser dialog, select a save location and enter a unique file name.
9. Click **OK**.

Configuring Sybase Control Center Logging for Performance Diagnostics

Change the logging behavior for Sybase Control Center (SCC) to better capture events for the adminstration framework.

Only enable SCC logging to diagnose performance. To enable logging:

1. Open `<SCC_HOME>\conf \log4j.properties`.
2. Edit following properties as desired:
   - `log4j.appender.agent.File`
   - `log4j.appender.agent.MaxFileSize`
   - `log4j.appender.agent.MaxBackupIndex`

If you need to diagnose SCC performance issues, review performance data with the `<SCC_HOME>\log\executionTime.log`:

1. Open `<UnwiredPlatform_InstallDir>\SCC-XX\plugins \com.sybase.supadminplugin\agent-plugin.xml`.

2. Add the following line to the file under the `<properties>` element:

```
<set-property property="log_MO_method_execution_time"
value="enable_log_mo_method_execution_time" />
```

3. Open *<UnwiredPlatform_InstallDir>*`\`*SCC-XX*`\conf\log4j.properties`.
4. If you are experiencing log truncation issues, edit the following lines to change the default values for maximum file size (default: 5MB) and maximum backup index (default: 10 files) to the values shown in this example:

```
## file appender (size-based rolling)
log4j.appender.executionTime=org.apache.log4j.RollingFileAppender
log4j.appender.executionTime.File=${com.sybase.ua.home}/log/
executionTime.log
log4j.appender.executionTime.layout=org.apache.log4j.PatternLayou
t
log4j.appender.executionTime.layout.ConversionPattern=%d [%-5p]
[%t] %c.%M(%L) - %m%n
log4j.appender.executionTime.MaxFileSize=50MB
log4j.appender.executionTime.MaxBackupIndex=20
## log MO method execution time
log4j.logger.com.sybase.uep.sysadmin.management.aop=INFO,executio
nTime
```

5. Restart SCC.

   The `executionTime.log` file now appears in the *<UnwiredPlatform_InstallDir>*`\`*SCC-XX*`\log` folder.

Alternately, you can use the Adobe Flex log to track performance in Sybase Control Center:

1. Modify the *<UnwiredPlatform_InstallDir>*`\`*SCC-XX*`\plugins\com.sybase.supadminplugin\agent-plugin.xml` file as indicated in step 2, above.
2. Restart SCC.
3. Log in and perform your regular administrative tasks.
4. View the execution time indicators for these operations in the cookie file `supatcookie.sol`. The location of this file varies depending on your operating system:

Operating System	Location
Windows XP	`C:\Documents and Settings\<`*username*`>\Application Data\Macromedia\Flash Player\#SharedObjects`
Windows Vista	`C:\Users\<`*username*`>\AppData\Roaming\Macromedia\Flash Player\#SharedObjects`

Operating System	Location
Macintosh OS X	`/Users/<username>/Library/Preferences/Macromedia/Flash Player/#SharedObjects`
Linux	`/home/<username>/.macromedia/Flash_Player/#SharedObjects`

5. Analyze the log using your preferred method of data analysis.

See also

- *Configuring Server Log Settings* on page 249
- *Enabling and Configuring Domain Logging* on page 252
- *Configuring Messaging and Mobile Workflow Runtime Logging* on page 255
- *Configuring Messaging Device Logging* on page 256
- *Configuring RSOE Logging* on page 258
- *Configuring and Enabling Relay Server Logging* on page 258
- *Enabling Custom Log4j Logging* on page 259

Configuring Messaging and Mobile Workflow Runtime Logging

The level of detail in messaging synchronization logs is determined by the log level. You can view this information either in the server log, or you can open the logs files from the `<UnwiredPlatform_InstallDir>\Servers\MessagingServer\Trace` folder.

1. To include messaging runtime log information in the server logs, increase server log levels for the MSG component to capture events beyond the default level of INFO.

   The messaging components that log to the Unwired Server logs are:
   - SUPBridge – used to connect Unwired Server and the messaging synchronization subcomponent.
   - JMSBridge – used for JMS messaging synchronization.
   - MOLogSystem – used to enable system logging across all messaging subcomponents.
2. Alternately, you can manually edit *`UnwiredPlatform_InstallDir`*`\Servers\MessagingServer\Data\TraceConfig.xml`, to increase the default level of server logging. See *Configuring Mobile Workflow Tracing*.

See also

- *Runtime Message Tracing (TraceConfig.xml) Configuration File* on page 361
- *Configuring Server Log Settings* on page 249
- *Enabling and Configuring Domain Logging* on page 252
- *Configuring Sybase Control Center Logging for Performance Diagnostics* on page 253

- *Configuring Messaging Device Logging* on page 256
- *Configuring RSOE Logging* on page 258
- *Configuring and Enabling Relay Server Logging* on page 258
- *Enabling Custom Log4j Logging* on page 259

Configuring Mobile Workflow Tracing

Set the tracing level for mobile workflow client applications. When mobile workflow tracing is enabled, messages appear as .txt files in `<UnwiredPlatform_InstallDir>\UnwiredPlatform\Servers\UnwiredServer\logs\WorkflowClient`.

1. Open `<UnwiredPlatform_InstallDir>\UnwiredPlatform-XX\Servers\MessagingServer\Data\TraceConfig.xml`.
2. After the last line in the `<TraceConfig>` section, add:

   ```
   <Module Name="WorkflowClient" Level="<traceLevel>"
   Desc="<myWorkflowDescription>" />
   ```

   These are the available trace levels:

Level	Messages logged
All	Complete system information
Trace	Finer-grained informational events than debug
Debug	Very fine-grained system information, warnings, and all errors
Info	General system information, warnings, and all errors
Warn	Warnings and all errors
Error	Errors only
Off	Do not record any messages

3. Save the file.
4. Restart the server for the changes to take effect.

See also

- *Runtime Message Tracing (TraceConfig.xml) Configuration File* on page 361

Configuring Messaging Device Logging

By default, messaging device logs are enabled; however, you can configure their behavior from Sybase Control Center. You can retrieve these logs and review them as required.

You can configure logging for a single device or create a template and use it with multiple devices.

1. In the left navigation pane of Sybase Control Center, click the **Device Users** node.

2. To configure an existing device, click the **Devices** tab or to configure a template, click **Device Templates** tab.
3. Click **Properties** then select **Device Advanced**.
4. Enter advanced properties:

   - Debug Trace Level
   - Debug Trace Size
   - Device Log Items

   See *Sybase Control Center > Configure > Configuring Unwired Platform > Device Users > Device Templates > Messaging Device Advanced Properties*.

See also

- *Configuring Server Log Settings* on page 249
- *Enabling and Configuring Domain Logging* on page 252
- *Configuring Sybase Control Center Logging for Performance Diagnostics* on page 253
- *Configuring Messaging and Mobile Workflow Runtime Logging* on page 255
- *Configuring RSOE Logging* on page 258
- *Configuring and Enabling Relay Server Logging* on page 258
- *Enabling Custom Log4j Logging* on page 259

Retrieving Device Logs

Send a request to Unwired Server to retrieve log files from a messaging-based synchronization (MBS) device.

Log file retrieval is supported only for messaging devices.

1. In the left navigation pane of Sybase Control Center, select **Device Users**.
2. In the right administration pane, click the **Devices** tab, and select **MBS** to view messaging-based synchronization devices.
3. Select a device, and click **Get Trace**.

   The status of the device must be either "online" or "offline" to retrieve the log.
4. Click **OK**.
5. When the device is online, check the device log. The default locations are `<UnwiredPlantform_InstallerDir>\Data\Messaging\ClientTrace` folder of the CDB node for cluster, or `<UnwiredPlantform_InstallerDir>\UnwiredPlatform\Servers\MessagingServer\Data\ClientTrace` folder for a single node.

   Trace files, which can be either `.log` or `.txt` file types, have the following file name structure: *yyyyMMddHHmmss_UserName_DeviceName_FileName*. For example: `20091217103050_User1_Emulator67215793_Messaging_xce.log`.

Configuring RSOE Logging

By default, the relay server outbound enabler (RSOE) is configured to log only errors, which is sufficient in a production environment. Increase the log level if you require additional detail to troubleshoot the RSOE.

You configure RSOE logging when you set up or start an RSOE. Both configuration and startup features are available on the RSOE tab by clicking **Servers** > ***ServerName*** > **Server Configuration**.

See also

- *Chapter 5, Relay Server Clusters* on page 53
- *Relay Server Documentation* on page 54
- *Relay Server Configuration Files* on page 363
- *Relay Server Utilities* on page 329
- *Configuring Server Log Settings* on page 249
- *Enabling and Configuring Domain Logging* on page 252
- *Configuring Sybase Control Center Logging for Performance Diagnostics* on page 253
- *Configuring Messaging and Mobile Workflow Runtime Logging* on page 255
- *Configuring Messaging Device Logging* on page 256
- *Configuring and Enabling Relay Server Logging* on page 258
- *Enabling Custom Log4j Logging* on page 259

Configuring and Enabling Relay Server Logging

By default, the relay server is configured to log only errors, which is sufficient in a production environment. Increase the log level if you require more detail to troubleshoot the relay server.

Errors appear regardless of the log level specified, and warnings appear only if the log level is greater than 0.

1. Open `rs.config`.

   The location of this file varies depending on the type of Web server used to host relay server. See *Relay Server (rs.config) Configuration File*.
2. Locate the `[options]` section.
3. Set these properties:

   - `start=no`
   - `verbosity=0`

     Edit the value to 1, 2, 3, or 4. The higher the number, the higher the degree of log event verbosity. For troubleshooting and maintenance, levels 1 or 2 should be sufficient.
4. Ensure the relay service is running, then run:

   rshost -f rs.config -u -q -qc

5. Check the relay server desktop tray icon to ensure there are no errors.
   - If there are no errors, close the command window.
   - If there are errors, check the `rs.config` file for errors and try again.
6. Validate the setup by opening the log file and confirming that the log entries reflect the log level you configure.

See also
- *Relay Server (rs.config) Configuration File* on page 364
- *Configuring Server Log Settings* on page 249
- *Enabling and Configuring Domain Logging* on page 252
- *Configuring Sybase Control Center Logging for Performance Diagnostics* on page 253
- *Configuring Messaging and Mobile Workflow Runtime Logging* on page 255
- *Configuring Messaging Device Logging* on page 256
- *Configuring RSOE Logging* on page 258
- *Enabling Custom Log4j Logging* on page 259

Enabling Custom Log4j Logging
Use any editor (text, XML, or an IDE) to create a `log4j.xml` file.

Prerequisites
Understand the use restrictions for log4j logging.

Note: The file format of this file falls outside the scope of this document. For details about `log4j.xml` format, see *http://logging.apache.org/log4j/*.

Task

1. In an editor, create a `log4j.xml` file.
2. Define the appenders you require and save the file. Appenders must be named, have a class defined, and require one or more parameters.
3. Add the file to all servers in your cluster. The file must be saved in: `<UnwiredPlatform_InstallDir>\Servers\UnwiredServer\Repository\`.
4. Repeat these steps on each server in the cluster.
5. Restart all servers.

See also
- *Configuring Server Log Settings* on page 249
- *Enabling and Configuring Domain Logging* on page 252
- *Configuring Sybase Control Center Logging for Performance Diagnostics* on page 253

- *Configuring Messaging and Mobile Workflow Runtime Logging* on page 255
- *Configuring Messaging Device Logging* on page 256
- *Configuring RSOE Logging* on page 258
- *Configuring and Enabling Relay Server Logging* on page 258

Log4j Restrictions

The default Unwired Server runtime logging and log4j logging are peer systems; the implementation of one does not usually impact the other, unless functionality overlaps.

Border conditions present in the `log4j.xml` configuration file may interfere with the configuration defined by `<UnwiredPlatform_InstallDir>\Servers\UnwiredServer\Repository\logging-configuration.xml`. Do not create appenders that output to:

- Any console (either System.out or System.err) with any appender, including ConsoleAppender. Log data destined to these appenders is ignored by Unwired Platform runtime components.
- The Unwired Server log. By default you can find the log file is created as `<UnwiredPlatform_InstallDir>\Servers\UnwiredServer\logs\<hostname>-server.log`.

Any changes you make to a `log4j.xml` configuration are applied only to the node on which the change is made; they are not propagated across the entire cluster. Therefore, you must edit the file on each node individually, as required.

CHAPTER 12 Operation Maintenance

Operation maintenance includes the regular administration activities that keep your mobile enterprise from running properly.

Operation maintenance includes updating, changing, or repairing various components already deployed to an Unwired Platform deployment. Platform administrators typically performed these activities on a regular or semi-regular schedule, often during non-peak usage hours, to keep the environment running smoothly.

Runtime Maintenance Cleanup

An important part of maintaining system health is periodically cleaning up the runtime environment. Over time, runtime artifacts accumulate and can impact system performance.

Sybase Unwired Platform lets the SUP Administrator and SUP Domain Administrator automate cleanup of accumulated data items at the domain level. Types of data that are eligible for cleanup include: synchronization cache, online cache, client log, error history, and subscriptions.

Set up a schedule for each of these categories, so the clean up processes run when system usage is low. You can also manually purge accumulated data items at any time. You can use the Administration API to automate clean up at the package level.

Sybase also recommends manual cluster-level cleanup for unused devices and users.

See also

- *Backup and Recovery* on page 264
- *Sybase Unwired Platform Licenses* on page 274

Scheduling Domain-Level Cleanup

Periodically clean up accumulated data maintenance items in cache that are no longer needed.

You can automate domain-level cleanup based on a configured schedule for specific cleanup categories.

Running the cleanup options uses system resources, so Sybase recommends that you schedule these tasks when system load is lightest. Optionally you can run the cleanup tasks manually.

1. In the Sybase Control Center left navigation pane, expand the **Domains** tab and select a domain.
2. In the right pane, select the **Scheduled Task** tab.

3. Under Task, select one of the options you want to schedule, and then select **Properties** to set up its automatic schedule:

Option	Description
Subscription Cleanup	Removes subscriptions that are no longer referenced by any active users. • Replication-based synchronization – removes subscriptions not used since the last synchronization. • Message-based synchronization – removes subscriptions if Unwired Server has not received a synchronization message since the given date.
Error History Cleanup	Removes historical data on MBO data refresh and operation replay failures, which result from system or application failures. System failures may include network problems, credential issues, and back-end system failure. Application failures may include invalid values, and non-unique data. **Note:** Only error messages are removed.
Client Log Cleanup	Removes client log records that have already been synchronized to the device, or are no longer associated with active users.
Synchronization Cache Cleanup	Removes logically deleted rows in the cache that are older than the oldest synchronization time on record in the system. Synchronization activity for all clients establish the oldest synchronization time. This cleanup task also removes unused or stale partitions.

4. Select **Enable**. Schedules run until you disable them, or they expire.

See also

- *Purging Unused Devices* on page 262
- *Purging Unused Device Users* on page 263
- *Maintaining Platform Databases* on page 263

Purging Unused Devices

Periodically clean up devices not used since a given date. This purges device associated items such as subscriptions.

1. In the left navigation pane, select the cluster, then **Device Users**.
2. In the right pane, select the **Devices** tab, and click **Purge**.
3. Specify the number of inactive days to use as the criteria for purging devices, for example, 90. Any devices that have not been used during that time period are purged.

Use a number of days that is large enough to account for periods of time during which users may not be active users.

4. Click **OK** to purge devices, or **Cancel** to end.

See also

- *Scheduling Domain-Level Cleanup* on page 261
- *Purging Unused Device Users* on page 263
- *Maintaining Platform Databases* on page 263

Purging Unused Device Users

Periodically clean up users. User cleanup removes any user-specific data stored in the consolidated database, such as personalization keys. If you remove a security configuration, all users registered with that security configuration are also removed.

1. In the left navigation pane, select the cluster, then **Device Users**.
2. In the right pane, select the **Users** tab, and click **Purge**.
3. In the Purge Users dialog, specify:
   - **Security configuration** – select a configuration, for example, admin.
   - **Number of days since last authentication** – specify the number of days.
4. Click **OK** to purge users that meet the criteria.

See also

- *Scheduling Domain-Level Cleanup* on page 261
- *Purging Unused Devices* on page 262
- *Maintaining Platform Databases* on page 263

Maintaining Platform Databases

Platform databases need to be rebuilt regularly. Without regular maintenance, the database log can grow to be large (several gigabytes in size), and performance could degrade.

If you experience long shutdown times with data services for the Unwired Platform data tier, you need to unload and reload data the dbunload utility. This rebuilds your database and maintains a healthy data tier performance.

To avoid this issue, Sybase recommends that you perform this action every four or five months.

1. Stop all runtime services (data tier and server nodes), including the consolidated database, the cluster database, and the messaging database, and synchronization services.
2. Perform a full off-line backup of each, by copying the database and transaction log files to a secure location. See *Backing Up SQL Anywhere Databases*.

3. Rebuild each runtime database with the SQL Anywhere Unload (dbunload) utility.

   The dbunload utility is available in the *UnwiredPlatform_InstallDir* `\Servers\SQLAnywhere11\BIN32` directory.
4. Validate the data before restarting the services.

See also

- *Sample Backup and Recovery Plan* on page 265
- *Scheduling Domain-Level Cleanup* on page 261
- *Purging Unused Devices* on page 262
- *Purging Unused Device Users* on page 263

Control Transaction Log Size

Control the size of transaction logs to prevent the log files from growing indefinitely.

You could use the SQL Anywhere **dbbackup** utility, with the **-xo** flag. The **-xo** flag deletes the current transaction log file, once it has been backed up successfully, and creates a new one. See *SQL Anywhere® Server – Database Administration* for information.

You could also use a variant of the SQL Anywhere **BACKUP DATABASE** command. See the *SQL Anywhere Server – SQL Reference* for **BACKUP DATABASE** command options. This example performs daily backups automatically from within the database server:

```
CREATE EVENT NightlyBackup
SCHEDULE
START TIME '23:00' EVERY 24 HOURS
HANDLER
BEGIN
   DECLARE dest LONG VARCHAR;
   DECLARE day_name CHAR(20);

   SET day_name = DATENAME( WEEKDAY, CURRENT DATE );
   SET dest = 'd:\\backups\\' || day_name;
            BACKUP DATABASE DIRECTORY dest
   TRANSACTION LOG RENAME;
END;
```

Backup and Recovery

Backup and recovery strategies are part of a larger availability and resiliency strategy you created when designing your network for Unwired Platform environment.

- Availability describes the degree of healthy system operations you can maintain under adverse conditions (for example, if multiple nodes in an Unwired Server cluster become unavailable due to a hardware failure).
- Resiliency describes how quickly healthy system operations return after service degradation.

Backup and recovery strategies can be one of two types: error correction and disaster recovery. The Unwired Platform documentation discusses error correction. For disaster recovery, you may need to engage with other vendors that provide solutions geared to maintaining the viability of your entire enterprise.

See also

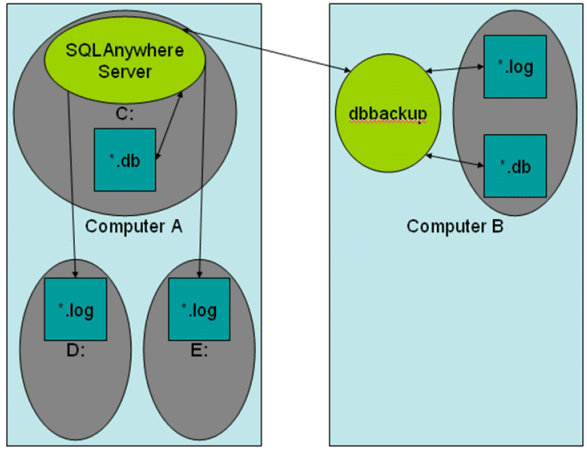
- *Data Management Overview* on page 189
- *Data Tier Clusters* on page 11
- *Runtime Maintenance Cleanup* on page 261
- *Sybase Unwired Platform Licenses* on page 274

Sample Backup and Recovery Plan

Provides a basic backup and recovery plan.

This diagram shows the architecture for a reasonably reliable backup and recovery strategy. Only Sybase Unwired Platform components related to database recovery are included.

Figure 1: Sample backup and recovery plan



Shown in the diagram:

- Computer A is where the SQL Anywhere database server is installed and runs under Sybase Unwired Platform. There are three physical disks on this computer:
  - The C: drive has the SQL Anywhere server and the database files (`default.db`, `clusterdb.db`, and `monitor.db`), which hold critical data that Sybase Unwired Platform requires to function.
  - The D: and E: drives hold identical (that is, mirrored) versions of the transaction log files (`default.log`, and `clusterdb.log`). Using SQL Anywhere terminology,

the D: drive holds the regular transaction log, and the E: drive holds the mirrored transaction log.

- Computer B is for long term backup, and requires only one drive (or backup tapes). Run the **dbbackup** utility from this computer periodically to obtain a full backup of the `*.db` and `*.log` files from Computer A.

See also

- *Using a Different Database and Log Path* on page 27
- *Maintaining Platform Databases* on page 263

Failure and Recovery Scenarios

Describes several failure scenarios (using the Sample Backup and Recovery Plan setup), how recovery works, and implications for Sybase Unwired Platform operation.

*Disk C has an unrecoverable error. The *.db files have been lost.*
Recovery:

1. Install a replacement disk, and use standard file restore procedures to reinstall the SQL Anywhere software, and whatever else is needed. If the restore returns the `*.db` files, there is no harm, but do not rely on these files to be valid. Instead, copy the last backup version of the `*.db` files across from Computer B.
2. If `*.log` files are healthy and have all the latest transactions recorded, run:

   ```
   dbeng11 <path>default.db -a <restored_path>default.log
   ```

   This command ensures that the restored database applies transactions from the current log to the older *.db file. Otherwise, the database does not start.
3. Start the SQL Anywhere server, which detects that the `*.db` files are not up-to-date with the checkpoints in the `*.log` files on drives D: and E: (which are unaffected by the C: drive failure). The server automatically replays transactions recorded in the transaction log to bring the database back to the state of all committed transactions at the time of the C: drive failure.

Sybase Unwired Platform can then start up and run normally. Sybase Unwired Platform mobile device clients are not affected except by the inability to synchronize between the time of the failure, and the time at which the recovery process has completed.

*Disk D: or E: failure. One of the *.log files has been lost.*
Recovery: Install a replacement disk and restore from backups.

Once the disk has been restored, copy the `*.log` files from the drive that did not fail to the one that failed. Restart the failed drive.

Complete failure of Computer A, and disks lost.
Recovery: See Restore the Consolidate Database for instructions. This should be a very infrequent event.

In this scenario, the database has lost all transactions since the previous backup to Computer B. Any Sybase Unwired Platform mobile device clients that synchronized between the time of the previous backup and the time of the failure cannot now sync. Clients must delete their remote device client database and start fresh. Any pending operations on these clients are lost. Clients that have not synchronized since the previous backup are unaffected.

Backing Up the File System

Regularly perform backups of the Sybase Unwired Platform files and directories.

Avoid backing up individual artifacts. Instead, Sybase recommends that you perform regular backups of entire directories as part of your disk backup schedule.

Note: At the same time the folder and disk backup is performed, update the Windows registry.

1. Plan the frequency of the file system backups to coincide with any changes made to the system, including:
   - Metadata changes (such as deployment of new mobile business object packages to the server)
   - Configuration changes (such as new enterprise information system connection)
2. Back up application artifacts, by using Sybase Control Center to export packages. This preserves each deployed package in an alternate location. See *Sybase Control Center online help > Manage > Managing Unwired Platform > Routine Command and Control Actions > Deploy > Exporting Package Content*.
3. Back up the contents in these Unwired Server directories:
   - `<UnwiredPlatform_InstallDir>\Servers\UnwiredServer\bin`
   - `<UnwiredPlatform_InstallDir>\Servers\UnwiredServer\config`
   - `<UnwiredPlatform_InstallDir>\Servers\UnwiredServer\logs`
   - `<UnwiredPlatform_InstallDir>\Servers\UnwiredServer\Repository`
4. Back up the contents of your system servers:
   - `<UnwiredPlatform_InstallDir>\Servers\Advantage910`
   - `<UnwiredPlatform_InstallDir>\Servers\MessagingServer`
   - `<UnwiredPlatform_InstallDir>\Servers\SQLAnywhere11`
5. Back up the contents of these Sybase Control Center directories:
   - `<UnwiredPlatform_InstallDir>\SCC-X_X\conf`

- `<UnwiredPlatform_InstallDir>\SCC-X_X\services\repository\repository.db`
- `<UnwiredPlatform_InstallDir>\SCC-X_X\log`

Next
To maintain a consistent backup state, Sybase recommends you back up the data consolidated database at these times as well. See the next task (Backing Up System Data) in this sequence.

Backing Up System Data

For platform data, back up Unwired Server runtime databases and Sybase Control Center (SCC) repositories using the process described for SQL Anywhere databases. Messaging database requires its own process.

Table 22. Runtime Database Default File Locations

Runtime Databases	Default File Locations

Unwired Server	For a Developer Edition installation, database files and transaction logs are installed:
	`<UnwiredPlatform_InstallDir>\UnwiredPlatform\Servers\UnwiredServer\data`
	• Consolidated database:
	  • Database file: `default.db`
	  • Transaction log: `default.log`
	• Cluster database:
	  • Database file: `clusterdb.db`
	  • Transaction log: `clusterdb.log`
	• Monitor database:
	  • Database file: `monitordb.db`
	  • Transaction log: `monitordb.log`
	For a separate Data-tier node, these database files and transaction logs are installed:
	• With a Microsoft cluster, in the Microsoft cluster folder you created: `<Microsoft_cluster_folder>\CDB`
	• With no Microsoft cluster: `<UnwiredPlatform_InstallDir>\UnwiredPlatform\Data\CDB`

Runtime Databases	Default File Locations
Sybase Control Center	Database files and transaction logs on each Unwired Server node: `<UnwiredPlatform_InstallDir>\SCC-3_0\services\Repository` • Database file: `scc_repository.db` • Transaction log: `scc_repository.log`

Note: When you make a backup, decide where to store the backup files: on the Unwired Server host or on some other computer or third-party hardware/software package used for backup purposes.

Backing Up SQL Anywhere Databases

Back up platform data warehoused in SQL databases using SQL Anywhere utilities.

Prerequisites

On backup hosts, verify that SQL Anywhere software is installed, and the PATH is set. If you are running Sybase Unwired Platform as a cluster, SQL Anywhere is installed under the Sybase Unwired Platform installation directory. Otherwise, install another copy of Sybase Unwired Platform on backup machines. Otherwise, SQL Anywhere components may not work as expected.

Task

By default, Unwired Platform uses SQL Anywhere for the consolidated and cluster databases. `clusterdb` is present if Unwired Server has been configured to run as a cluster. Use various SQL Anywhere utilities to help you with these tasks:

- **dbvalid** – validates the integrity of the database by checking the index keys on some or all of the tables in a database, as well as on the database file itself.
- **dblocate** – locates any SQL Anywhere database servers (consolidated or cluster) running over TCP/IP on the immediate network, and prints a list of the database servers and their addresses. This list includes alternate server names. Depending on your network, it may take several seconds for **dblocate** to print its results.
- **dbbackup** – makes a backup copy of all files for a single database from a remote backup location. A simple database consists of two files: the main database file and the transaction log. If you mirrored the transaction log you need not back up this file. All backup file names are identical to database file names. The image backup contains a separate file for each backed-up file.
- **dblog** – mirrors the transaction log used for these databases.

Backing Up Database Files

Regularly perform a remote backup of data so it can be recovered. Before you back up system data, ensure the data is valid.

Note: Depending on the design of your production environment, the **dbvalid** and **dbbackup** commands may not work remotely.

The database server is named `<clustername>_primary`.

---

1. Validate the databases:
   a) Ensure there are no active connections to the server.
   b) Validate the consolidated database:

      ```
      dbvalid.exe -c "DBF=default.db;UID=dba;PWD=SQL"
      ```

   c) Validate the cluster database:

      ```
      dbvalid.exe -c "DBF=clusterdb.db;UID=dba;PWD=SQL"
      ```

   d) Validate the monitoring database:

      ```
      dbvalid.exe -c "DBF=monitor.db;UID=dba;PWD=SQL"
      ```

2. On the backup machine, verify that SQL Anywhere software is installed, and the PATH is set.
3. Back up the databases to archive the system data:
   - For consolidated databases, run:

     ```
     dbbackup -c
     "ENG=<clusterName>_primary;DBN=default;UID=dba;PWD=SQL"
     \SQLAnybackup
     ```

   - For cluster databases, run:

     ```
     dbbackup -c
     "ENG=<clusterName>_primary;DBN=clusterdb;UID=dba;PWD=SQL"
     \SQLAnybackup
     ```

   - For monitor database, run:

     ```
     dbbackup -c
     "ENG=<clusterName>_primary;DBN=monitor;UID=dba;PWD=SQL"
     \SQLAnybackup
     ```

   This creates `default.db`, `default.log`, `clusterdb.db`, and `clusterdb.log`, and `monitor.db`, `monitor.log` in the `\SQLAnybackup` directory on the backup computer.
4. As a precaution, validate the backups are suitable for recovery:
   a) On the backup computer, create a temporary working directory (such as `\tmp`).
   b) Under the temporary directory, create an identical directory structure for the two log locations. You may need to use the **subst** command to map local directories to drive letters used on the runtime computers to the backup location.
   c) Copy `*.log` to these locations.

d) Run **dbvalid** on the `\tmp` copy of the `.db` file.

WARNING: Do not run **dbvalid** on the backup copy itself (in the `\SQLAnybackup` directory of this example). The command runs, but corrupts your `.db` file so it cannot be used in recovery.

- If validation succeeds, the backup in `\SQLAnybackup` can be used for recovery; delete the files in the `\tmp` and log directories.
- If validation fails, the backup is not usable for recovery; try again.

Next

With the archive of the database complete, you can optionally back up the archive to a tape drive.

Backing up Data and Mirroring Transaction Logs

Mirror logs to make identical copies elsewhere on your network using SQL Anywhere utilities. Mirrored logs ensure runtime data is available and recoverable in the event of failure.

Use dblog command utility to set up log mirroring. Sybase recommends an extension of `*.mlg` for the mirrored log so the mirrored log can quickly be identified.

Mirror the transaction log, so that if database file becomes unrecoverable, you can use record of changes since the last backup in the transaction log to rebuild the database.

From the `<UnwiredPlatform_InstallDir>\Servers\SQLAnywhere11\BIN32` directory run the dblog command with the -m option: For example, to mirror the original log file on drive D:\ to drive E:\ use:

```
dblog -t D:\logs\default.log -m E:\logs\default_mirror.mlg
default.db
```

Backing Up Messaging Data

Use the **adsbackup** utility to back up messaging data. You can then use MORecover to recover the data to an existing database.

The backup target can be any folder. A collection of files constituting the backed-up data is created in a folder you specify.

Note: Backup files only include raw data in the original tables, without the index data. Therefore, they cannot be used directly as a database and must be recovered with MORecover before they can be used.

You can schedule this command to run regularly when your system is lightly loaded, since the backup operation runs simultaneously with the messaging server's normal processing.

1. Change to `<UnwiredPlatform_InstallDir>\Servers\AdvantageXXX\Server`.
2. Run the **adsbackup** utility to create a MORecover snapshot; for example:

   ```
   adsbackup.exe
   iAPPLICATIONS,CFG_IDS,CFG_PROP_VALUES,CFG_SUBFOLDER_PROP_V
   ```

```
ALUES,CFG_TEMPLATES,DEVICES,USER_DEVICE,USERS "C:\Sybase
\UnwiredPlatform-X_X\Servers\MessagingServer\Data\OBR
\OBR.add" <MYBackupTarget>
```

See also

- *Advantage Database Server Backup (adsbackup) Utility* on page 350

Restoration of the Installation File System

Restore the Sybase Unwired Platform installation file system from a backup.

To perform a normal restoration, use the file or disk backup utilities used to perform the backup. Sybase recommends that you save the current installation directory before you restore from backup.

Note: You may also need to restore the Windows registry from the backup done at the same time.

Restoration of the Runtime Database

Restore the Sybase Unwired Platform runtime databases and transaction logs from a backup.

As discussed in *Failure and Recovery Scenarios*, if only one of C:, D:, or E: drives fail, recovery should be automatic once you have completed the appropriate tasks.

These steps are required in case of complete failure of all three drives:

1. Restore the computer's C:, D:, and E: drives from backup.
2. Delete the `*.db`, `*.log` files from their normal places after you have restored the file system.
3. Copy the `*.db` and `*.log` files from Computer B's backup directory to the normal locations on Computer A.

   Note: Copy the *.log files twice—once to the normal transaction logs directory, and once to the mirrored transaction logs directory.
4. Restart Sybase Unwired Platform.
5. If there have been package deployments or other cluster-affecting operations since the last database backups, the file-system data corresponding to the packages may be out-of-sync with the database contents related to these packages. If this has occurred, the Sybase Unwired Platform servers cannot fully start. Check server bootstrap log and the <hostname>-server.log file. These logs include server mismatch messages that prevents server from starting normally. To recover from this:
   a. Choose one of the Unwired Servers.
   b. Shut down that server.
   c. Edit the `sup.properties`, and change the `cluster.version` property value to match that of the current cluster as reported in the logs. This file is located in

`<UnwiredPlatform_InstallDir>\Servers\UnwiredServer\Repository\Instance\com\sybase\sup\server\SUPServer`.

d. Run `updateProps -r` from the same directory to apply the change into the `clusterdb`.

e. Restart that Unwired Server.

Note: This server should be able to take over as the Sybase Unwired Platform primary. Sybase recommends you redeploy all your packages (using the **UPDATE** option) to make sure all the packages you expect to be available really are. All of your Sybase Unwired Platform clients should delete their UltraLite databases, and perform full synchronizations.

Restoration of the Messaging Data

Restore the Sybase Unwired Platform messaging data tables from a backup by running the MORecover utility.

You can find this utility in the `<UnwiredPlatform_InstallDir>\Servers\Messaging\Server\Bin` folder.

Before running MORecover to restore your database, convert the data files created by **adsbackup** back to a standard database format, by running **adsbackup** with the -r option. You can then use MORecover to restore the backed-up data.

For example, if you sent your backed-up data output to `c:\bak`, but restored the data to `c:\bak\restore`, the command line syntax for **adsbackup** is:

```
adsbackup.exe -r "c:\bak\obr.add" "c:\bak\restore\obr.add"
```

When you run the MORecover utility, use the restore location. For example:

```
MORecover "c:\bak\restore\obr.add"
```

See also

- *Unwired Server Database File Recovery (MORecover) Utility* on page 352

Sybase Unwired Platform Licenses

Sybase Unwired Platform is available in three editions.

Edition	Summary
Personal Developer Edition	• Includes Unwired Server, data tier, and Sybase Unwired WorkSpace development tools. • Allows use in development systems and development-test systems only; not for use in production systems. • Requires all Unwired Platform components to be installed on the same host. • Allows a maximum of five client devices.
Enterprise Developer Edition	• Includes Unwired Server, data tier, and Sybase Unwired WorkSpace development tools. • Allows use in development systems and development-test systems only; not for use in production systems. • Allows each installable component to be located on a separate host. • Allows clustered systems. • Allows a maximum of 20 client devices.
Enterprise Edition	• Includes Unwired Server and data tier components only. • Allows use in production systems and production-test systems only; not for use in development systems. • Allows each installable component to be located on a separate host. • Allows clustered systems. • Requires separate license for client devices.

See also

- *Runtime Maintenance Cleanup* on page 261
- *Backup and Recovery* on page 264

Cluster License Coordination

In a cluster, each server deployed to the environment must be licensed. Multiple servers cannot share a single license. However, all server nodes in the cluster can share device connection licenses.

In a clustered environment, you must use a license server so it can coordinate licensing requirements among all installed components:

- Server validation – each time a server starts, it connects and registers with the license server to check if there is a valid license for it. If there is a free license available, the server checks out the license and continues with the start-up process. If the number of licensed servers cannot be retrieved from the license file, or the license server confirms that a server is not licensed, Unwired Server stops.
- Device connection validation – because available device licenses are shared among all servers in the cluster, all connections to all servers must be accounted for. The cluster name enumerates each device connection made across clustered servers. Every server then checks out all device licenses when the servers start.

License Validation

Attributes in the license file control the base number of devices that can be registered, the number of servers (typically for clustered production environments) you install, and expiry dates for both devices and servers. The mechanism that counts device licenses varies, depending on your model.

There are two licensing models you can use with Unwired Platform:

- Unserved (local) license – uses a local license file for each Unwired Platform installation.
- Served (SySAM license server) – uses a SySAM license server to support multiple Unwired Platform installations.

For both models, Unwired Server always tracks available licenses and expiry dates, and writes license errors to the Unwired Server log. Administrators can always check these limits and take appropriate action when that limit is reached.

Unserved Model

In an unserved license model, licenses are validated at several intervals:

- At start-up – if Unwired Server cannot retrieve the number of licensed servers from the license file, or if the server is not licensed, Unwired Server stops.
- At device connection – when the device user tries to connect to Unwired Server, Unwired Server checks the device ID against the data tier. If the device falls within the device license limit, the device connection continues and operations proceed normally for both replication and messaging applications. If the device falls outside the limit, Unwired Server throws a license check exception to the client. See *System Administration Guide > Operations Maintenance > Platform Licenses > Device User License Limits*.
- Upon license expiry – if the date in the license file matches the current date, the license expires; Unwired Server generates a license expired error. The error text varies, depending on whether the server or the client connection licenses have expired. If a server license is expired, Unwired Servers also stop.

Served Model

In a served license model, licenses are validated at these intervals:

- At start-up – if Unwired Server cannot retrieve the number of licensed servers from the license file, or if the server is not licensed, Unwired Server stops.
- With each synchronization – the procedure varies slightly depending on the synchronization model used on the client:
  - For replication-based synchronization – after the device user is authenticated, Unwired Server uses the device ID to check the license into the data tier. If the device falls within the device license limit, synchronization proceeds. If the device falls outside the limit, Unwired Server throws a license check exception to the client.

    Administrators must monitor licenses carefully; there may be many devices connected to the server, but fewer licenses being used. See *System Administration Guide > Operations Maintenance > Platform Licenses > Device User License Limits*.
  - For messaging-based synchronization – when the device user tries to connect, Unwired Server checks the device ID against the data tier. If the device is registered, and the total number of devices registered falls within the device license limit, the message is processed normally. If the device is not registered, or the total number of devices registered falls outside the limit, Unwired Server throws a license check exception to the client.
- Upon license expiry – if the license expires, Unwired Server generates a license expired error. The error varies, depending on whether the server or the client connection licenses have expired. When a server license expires, Unwired Servers also stop.

Device User License Limits

Licenses limit not just how many components you can install on a network, but how many users can connect to your servers.

If you notice messages similar to these errors in the Unwired Server log, then connection requests from registered users have exceeded the licensed limit:

```
2009-12-28 18:01:59.872 INFO    MMS          Thread-19
[com.sybase.sup.server.lm.LicenseUtil] The number of registered
devices has reached the maximum limit for your license.
2009-12-28 18:01:59.965 INFO    MMS          Thread-19
[com.sybase.djc.mobilink.LoginHandler] The number of registered
users has reached the maximum limit for your license.
2009-12-28 18:02:00.168 ERROR   Mobilink     Thread-19
[com.sybase.ml.sup.Logger] [-10052] authenticate_parameters scripts
return 4000
```

For example, a trial license limits you to only five device users. If a sixth user tries to connect, the error is logged accordingly.

In cases where the number of users in your environment exceeds that of your license, you can either:

- Upgrade your license and manually change the license in your environment.
- Control the number of device user connections at a given moment in SCC. For example, you can view the total number of users in the User Statistics tab of the Monitor node.

If the number of device users is too high for your license, you can manually delete unused devices from the system in the Devices node. See *Sybase Control Center online help > Manage > Managing Unwired Platform > Routine Command and Control Actions > Provision > Device Users > Devices > Deleting Replication and Messaging Devices*.

Checking System Licensing Information

Review licensing information to monitor available and used device licenses, license expiry dates, and other license details. This information allows administrators to manage license use and determine whether old or unused device licenses should be tranferred to new devices.

Unwired Platform licensing is configured during installation. However, if necessary, license details can be changed at a later time. See *Manually Updating and Upgrading License Files*.

1. In the left navigation pane, select the top-level tree node.
2. In the right administration pane, select the **General** tab, and click **Licensing**.
3. Review the following licensing information:
   - Production edition – the edition of the software you have installed.
   - Total device license count – the total number of device licenses available with your license. This count limits how many devices can connect to your servers. See *Device User License Limits*.
   - Used device license count – the total number of licenses currently attached to devices. If all of your available device licenses are in use, you can either upgrade your license or manually delete unused devices to make room for new users. See *Device User License Limits*.
   - Device license expiry date – the date and time at which the device license expires. When a device license expires, Unwired Server generates a license expired error and connection requests from registered devices are unsuccessful.
   - Server license expiry date – the date and time at which the server license expires. When a server license expires, Unwired Server generates a license expired error and Unwired Server is stopped.
   - Server license type – the type of license currently used by Unwired Platform. For more information on license types, see *Platform Licenses*.
   - Overdraft mode – allows you to generate additional licenses in excess of the quantity of licenses you actually purchased. This enables you to exceed your purchased quantity of licenses in a peak usage period without impacting your operation. This mode is either enabled or disabled, as specified by the terms of the agreement presented when you obtain such a license.
4. Click **Close**.

Manually Updating and Upgrading Licenses

You must upgrade Unwired Platform and Afaria licenses separately. For Unwired Platform upgrades, the procedure varies depending on whether you are using a served or unserved model.

To update or upgrade your license, ensure you have the updated or upgraded license file and save it in the appropriate location for your served or unserved license model. Then you need to run `license.bat` and restart Unwired Servers for license change to take affect.

1. *Updating and Upgrading Unwired Platform Licenses*

   The <UnwiredPlatform_InstallDir>\Servers\UnwiredServer\bin\license.bat script allows you to upgrade Unwired Platform licenses without having to re-run the installer. You can use this script in both served and unserved license models.

2. *Upgrading Afaria Licenses*

   If you have a version of Afaria installed with an earlier version of Unwired Platform, and need to upgrade to a standalone version of Afaria with more license options enabled, you need to change the serial number before they are enabled.

Updating and Upgrading Unwired Platform Licenses

The `<UnwiredPlatform_InstallDir>\Servers\UnwiredServer\bin\license.bat` script allows you to upgrade Unwired Platform licenses without having to re-run the installer. You can use this script in both served and unserved license models.

Prerequisites

For unserved models, ensure that you have already copied the new license file to the `<UnwiredPlatform_InstallDir>\Servers\UnwiredServer\licenses` folder.

Task

1. Use the license to locate information that the `license.bat` script requires. See *Locating Information in a License File*.

   When you set up your license model, Sybase license fulfillment generates a new Unwired Platform license that includes a new serial number. For subsequent upgrades, use the same serial number to incrementally update licenses.

2. From the command prompt, change to the `bin` folder for Unwired Server.
   For example, `cd C:Sybase\UnwiredPlatform\Servers\UnwiredServer\bin\`.

3. Run `license.bat`:

   `license.bat PE LT <licenseNumber>`,

where PE is the Production Edition, LT is the License Type, and <licenseNumber> are the number of licensed devices supported.

See also

- *License Upgrade (license) Utility* on page 344
- *Upgrading Afaria Licenses* on page 281

Locating Information in a License File

After you download a license file, you must extract some information from it to complete your installation. When you run the Unwired Platform installer, enter this information on the license details page.

1. Use a text editor to open your license file.
2. Locate the uncommented line that begins with the string for your Unwired Platform edition:
   - Enterprise Edition – INCREMENT SUP_ENTSRVR
   - Enterprise Developer Edition – INCREMENT SUP_ENTDEV
   - Personal Developer Edition – INCREMENT SUP_DEVEVELOPER

   For example:
   - Enterprise Edition would be similar to this.

     ```
     ...
     INCREMENT SUP_ENTSRVR SYBASE 2011.11150 permanent uncounted \
        VENDOR_STRING=PE=EE;LT=CP HOSTID=000c29d300bd
     PLATFORMS="i86_n \
     ...
     ```
   - Enterprise Developer Edition would be similar to this.

     ```
     ...
     INCREMENT SUP_ENTDEV SYBASE 2011.11150 permanent uncounted \
        VENDOR_STRING=PE=EE;LT=CP HOSTID=000c29d300bd
     PLATFORMS="i86_n \
     ...
     ```
   - Personal Developer Edition would be similar to this.

     ```
     ...
     INCREMENT SUP_DEVELOPER SYBASE 2011.11150 permanent uncounted \
        VENDOR_STRING=PE=EE;LT=CP HOSTID=000c29d300bd
     PLATFORMS="i86_n \
     ...
     ```

   The rest of the examples in this section show the beginning of this line as it would appear for Enterprise Edition. The details illustrated apply equally to all editions.
3. Determine whether the server license is served or unserved.

   If the line you located in step 2 ends with "uncounted" it is an unserved license. For example:

   ```
   ...
   INCREMENT SUP_ENTSRVR SYBASE 2011.11150 permanent uncounted \
   ```

```
   VENDOR_STRING=PE=EE;LT=CP HOSTID=000c29d300bd PLATFORMS="i86_n
\
...
```

If that line ends with a number immediately following a date, it is a served license. For example:

```
...
INCREMENT SUP_ENTSRVR SYBASE 2011.11150 permanent 10 \
   VENDOR_STRING=PE=EE;LT=CP HOSTID=000c29d300bd PLATFORMS="i86_n
\
...
```

4. Determine the product edition and license type for the license.

   For both served and unserved licenses, note the value of PE (product edition) and LT (license type) in the line following the line you located in step 2. For example:

   ```
   ...
   INCREMENT SUP_ENTSRVR SYBASE 2011.11150 permanent uncounted \
      VENDOR_STRING=PE=EE;LT=CP HOSTID=000c29d300bd PLATFORMS="i86_n
   \
   ...
   ```

   The PE value is the license product edition value; "EE" in the example above.

   The LT value is the license type value; "CP" in the example above.

5. If you are installing Enterprise Edition, determine the number of client licenses.

   If your license type is Development and Test (DT), you can change this number later.

   a) Locate the uncommented line, beginning with INCREMENT SUP_ENTCLIENT.

      For example:

      ```
      INCREMENT SUP_ENTCLIENT SYBASE 2011.11150 permanent uncounted \
               VENDOR_STRING=PE=EE;LT=ST HOSTID=000c29d300bd
      PLATFORMS="i86_n \
      ...
      ```

   b) Determine whether the client licenses are served or unserved.

      If the line beginning with INCREMENT SUP_ENTCLIENT ends with "uncounted" the client licenses are unserved. For example:

      ```
      INCREMENT SUP_ENTCLIENT SYBASE 2011.11150 permanent uncounted \
         VENDOR_STRING=PE=EE;LT=ST HOSTID=000c29d300bd
      PLATFORMS="i86_n \
         x64_n" ISSUER="CO=Sybase,
      Inc.;V=2.0;AS=A;MP=3120;CP=100;EGO=" \
      ...
      ```

      If that line ends with a number immediately after a date, the client licenses are served. For example:

      ```
      INCREMENT SUP_ENTCLIENT SYBASE 2011.11150 permanent 100 \
          VENDOR_STRING=PE=EE;LT=ST HOSTID=000c29d300bd
      PLATFORMS="i86_n \
      ...
      ```

c) Determine the number of client licenses.

For unserved client licenses, the number of client licenses is the value of CP two lines below the line beginning with INCREMENT SUP_ENTCLIENT. For example:

```
INCREMENT SUP_ENTCLIENT SYBASE 2011.11150 permanent uncounted \
    VENDOR_STRING=PE=EE;LT=ST HOSTID=000c29d300bd
PLATFORMS="i86_n \
    x64_n" ISSUER="CO=Sybase,
Inc.;V=2.0;AS=A;MP=3120;CP=100;EGO=" \
...
```

For served client licenses, the number of client licenses is the value at the end of the line beginning with INCREMENT SUP_ENTCLIENT. For example:

```
INCREMENT SUP_ENTCLIENT SYBASE 2011.11150 permanent 100 \
    VENDOR_STRING=PE=EE;LT=ST HOSTID=000c29d300bd
PLATFORMS="i86_n \
...
```

See also

- *License Upgrade (license) Utility* on page 344

Upgrading Afaria Licenses

If you have a version of Afaria installed with an earlier version of Unwired Platform, and need to upgrade to a standalone version of Afaria with more license options enabled, you need to change the serial number before they are enabled.

The server configuration area of the Afaria Administrator lets you define parameters for the Afaria Server, including new license data.

1. Obtain a new serial number.

   Sybase license fullfilment generates an Afaria license string that includes a new serial number. For subsequent option upgrades, use the same serial number to incrementally update licenses.

2. Define new software licenses in Afaria Administrator.

   The License Compliance page appears empty until you define licensing details. Once you have defined these software licenses, the page shows data for Client category, Manufacturer, Application, Size, Version, # (number) Purchased, Effective and Expiration dates, and any Notes you add. Initially, licenses are sorted by Client category, Manufacturer, then Application. See *Afaria Reference | Platform > Server Configuration > License Compliance*.

See also

- *Updating and Upgrading Unwired Platform Licenses* on page 278

CHAPTER 13 Customization with the Client API

Sybase Unwired Platform includes a Java API that opens the administration and configuration of Sybase Unwired Platform to Java client applications you create. By building a custom client with the administration client API, you can build custom application to support Sybase Unwired Platform administration features and functionality within an existing IT management infrastructure.

The client you create connects to Unwired Server via Sybase Control Center embedded management server and Sybase Unified Agent. For details, see *Developer Guide for Unwired Server Management API*.

The administration client API includes these administration interfaces:

Interface	Includes methods that
`SUPServer`	Command and control operations for an Unwired Server instance, for example start, stop, and ping.
`SUPCluster`	Manage cluster security, monitoring configuration and domain creation for a cluster instance, and so on.
`SUPDomain`	Manage domains, deploy packages to a domain, set security configurations for a domain, and so on.
`SUPPackage`	Configure packages by setting up subscriptions, configuring cache groups, configuring endpoint properties, and so on.
`SUPMobileBusinesObject`	View mobile business object properties, operations, errors, endpoints, and so on.
`SUPOperations`	View operation properties, errors, endpoints, and so on.
`SUPDeviceUser`	Create templates, configure device settings, manage device users, lock devices, and so on.
`SUPMonitor`	Perform monitoring functions like viewing histories, summaries, details, and performance data for various platform components, and export data as required.
`SUPServerLog`	View, filter, delete and refresh logs, configure appenders, and so on, for Unwired Server and its embedded services like replication and messaging synchronization.
`SUPDomainLog`	View, filter, delete and refresh, and so on, a domain logs instance.

Interface	Includes methods that
`SUPServerConfiguration`	Configure an Unwired Server instance, as well as its listeners. All methods of this interface, except the apple push notification-related properties are metadata-based.
`SUPSecurityConfiguration`	Create, manage, and configure a security configuration with at least one authentication provider. You can add other providers (authentication, authorization, attribution, and audit) as required.
`SUPMobileWorkflow`	Manage and configure deployed mobile workflow packages.

See also

- *Data Encryption Implementation* on page 139

Javadocs

The administration client API installation includes Javadocs. Use the Sybase Javadocs for your complete API reference.

As you review the contents of this document, ensure you review the reference details documented in the Javadoc delivered with this API. By default, Javadocs are installed to `<UnwiredPlatform_InstallDir>\Servers\UnwiredServer\AdminClientAPI\com.sybase.sup.adminapi\docs\api\index.html`.

The top left navigation pane lists all packages installed with Unwired Platform. The applicable documentation is available with `com.sybase.sup.admin.client` package. Click this link and navigate through the Javadoc as required.

CHAPTER 14 System Reference

To manage the system effectively, it is crucial to know about Unwired Platform subsystem components and how they fit together. This part outlines many important aspects of the system in quick reference format.

It covers the location of crucial system files and file systems, as well as other reference material that you might need when you are administering the Unwired Platform production environment: for example, logging details, configuration properties, and ports, service names, and processes used by each component.

Installation Directories

Review the Sybase Unwired Platform installation directories to ensure a successful installation.

- The following tables document only the high-level folder structure in a complete installation on a single server.
- In all installations, most of the directories listed have subfolders.
- In custom installations, including installations for cluster environments, not all of the subfolders are present.

By default, Sybase Unwired Platform is installed in the `C:\Sybase\UnwiredPlatform` directory. You may have specified a different location.

Table 23. Unwired Platform installation directory subfolders

Folder	Description
`_jvm`	Files for Java Virtual Machine used by uninstaller.
`sup20ebflogs`	The output location of log files created each time Unwired Platform `installebf.bat` file is used. Use these logs to troubleshoot issues with the EBF installer. This directory is present in upgrade installations of Unwired Platform 2.0.
`Eclipse`	Files supporting the Eclipse development environment. **Note:** Present in developer installations only.
`InstallLogs`	The output location of log files created each time Unwired Platform installer is used. Use these logs to troubleshoot installer issues.

Folder	Description
JDKx.x.x_x or JDKx.x.x_x-x64	Files used for version of JDK required by Unwired Platform. If the folder ends in -x64, this is the JDK for 64-bit operating systems in a production deployment environment.
scc_cert	Certificate file for Sybase Control Center.
Servers	Server components that make up Unwired Platform and its mobile middleware services.
Servers\Advantage910	Device management components that administer devices from Sybase Control Center. Includes online help.
Servers\MessagingServer	Synchronization components used for messaging-based synchronization.
Servers\SQLAnywhere11	Synchronization components used for replication-based synchronization. Frequently used folders include: • BINXX – for utilities you might use. • data – for database files used by Unwired Platform.
Servers\UnwiredServer	The application server used in an Unwired Platform mobility environment.
Servers\UnwiredServer\licenses	Location where Unwired Platform licenses are saved. Every time a license is updated, copy new licenses here.
ThirdParty	Location of required runtime files for other components integrated into the Unwired Platform environment. Contains the Free Download Terms PDF files from the Sybase Legal Department.
Uninstallers	The executable and supporting files used to uninstall Unwired Platform.
Unwired_WorkSpace	Executables and supporting files used by Unwired Workspace. **Note:** Present in developer installations only.
Util	Utilities that the installer executes to check and validate external information, such as third-party software installations, database information, and Windows account information.

Your Sybase Unwired Platform license includes Sybase Control Center, which, by default, is installed in C:\Sybase\SCC-3_0.

Note: If you have other Sybase products installed, you may have two different versions of Sybase Control Center. Unwired Platform requires 3.0, so only this directory structure is documented.

Table 24. Sybase Control Center 3.0 installation directory subfolders

Folder	Description
`auth`	Library files used for related services in SCC. For example, JAAS.
`bin`	Scripts you can use to start or stop components of the SCC management framework.
`common`	Required files shared by SCC components.
`conf`	Configuration files used for SCC, including security providers for administration logins.
`ldap`	The LDAP-related files for SCC.
`log`	Log files used by SCC and its console plug-ins used capture management framework events exclusively. No Unwired Platform data is captured here, except for administration logins.
`plugins`	Location for managed resource plug-ins, including one for Unwired Platform.
`rtlilb`	Runtime library files used by SCC.
`server`	Class and library files used by the management framework server.
`services`	Class and library files used by SCC services.
`shared`	Class and library files shared by SCC and its plug-ins.
`utility`	Various utilities used by SCC.

Port Number Reference

Components of Sybase Unwired Platform rely on communication ports for inter-process coordination, data transfer, and administrative access.

Unwired Server Ports

The following list identifies Unwired Server ports, and default port assignments.

Type	Default	Description
Administration	2000 2001 (secure)	Ports on which Unwired Server listens for IIOP requests. The secure administration port is disabled by default.
Data change notification (DCN)	8000 (HTTP) 8001 (HTTPS)	Ports on which Unwired Server listens for DCN requests.
Replication-based synchronization (RBS)	2480 2481 (secure)	Port used to synchronize data with mobile devices.
Messaging-based synchronization (MBS)	5001	Port used to synchronize data with mobile devices.
Messaging server administration	5100	Port used by the messaging service for Sybase messaging clients.

You can use Sybase Control Center (SCC) to change Unwired Server ports after installation.

Note: If there is a conflict for port 2480 or 2481, Unwired Server will not start. In that event, you will not be able to use SCC to modify those Unwired Server ports. To correct the problem, you must temporarily stop the service that uses the conflicting port, then start Unwired Server so you can change the port from the SCC console.

Data Tier Ports

The following list identifies data tier server ports, and default port assignments.

Type	Default	Description
Cluster database server	5300	SQL Anywhere® database server port
Consolidated database (CDB) server	5200	SQL Anywhere database server port
Messaging database server	6262	Advantage Database Server® port
Monitoring database server	5400	SQL Anywhere database server port

You can use Sybase Control Center to change data tier server ports after installation.

Sybase Control Center Ports

The following list identifies Sybase Control Center server ports, and default port assignments.

Type	Default	Description
RMI agent port	9999	Defined in: `<InstallDir>\SCC-*\services\RMI\service-config.xml`
JMS messaging service	2100	Defined in: `<InstallDir>\SCC-*\services\Messaging\service-config.xml`
SCC repository database	3638	Defined in: `<InstallDir>\SCC-*\services\SccSADataserver\service-config.xml`
Web container	8282 (HTTP) 8283 (HTTPS)	Defined in: `<InstallDir>\SCC-*\services\EmbeddedWebContainer\service-config.xml`

To change a Sybase Control Center server port, you must edit the XML configuration file that defines the port.

Relay Server Ports

The following list identifies relay server ports, and default port assignments.

Type	Default	Description
HTTP	80	Port on which relay server listens for HTTP requests
HTTPS	443	Port on which relay server listens for HTTPS requests

You can use Sybase Control Center to change relay server ports after installation.

Reserved Ports

The following list identifies Sybase Unwired Platform reserved ports.

Do not use reserved port numbers for any purpose.

Type	Number	Description
Reserved port	4343 5500 8002 27000	Ports reserved for internal use by Unwired Platform components.

Even if the installer does not detect a conflict, the Windows operating system may use additional ports in the 1024-65535 range at a later time. In that event, you may encounter intermittent problems starting Unwired Platform services.

Refer to Microsoft operating system documentation to learn how to reserve ports and prevent the operating system from attempting to use them.

Other Ports

The following lists identify significant ports that are not directly associated with a Sybase Unwired Platform component.

OpenDS

The following ports are used by OpenDS LDAP server, which is supplied only with Sybase Unwired Platform Personal Developer Edition and Enterprise Developer Edition.

Type	Default	Description
LDAP server	10389	Port on which OpenDS server listens for LDAP requests.
Administration	4444	Not used by Unwired Platform.

Note: To change the OpenDS LDAP request port:

- Edit the file `<InstallDir>\Servers\UnwiredServer\OpenDS\config\config.ldif` to change the value of the `ds-cfg-listen-port` property.
- Edit the file `<InstallDir>\SCC-*\conf\csi.properties` to change the value of the `CSI.loginModule.8.options.ProviderURL` property.

SySAM License Server

If you deploy Unwired Platform with the served license model, all Unwired Platform hosts must have access to the SySAM license server port.

Type	Default	Description
SySAM license server	27000	Port on which SySAM license server listens for requests.

Unwired Platform Windows Services

Unwired Platform Windows services run automatically on your host, with many starting up when the host computer is started or when the installation process finishes. Determine what services exist for each runtime component and what dependencies exist among these services.

Depending on the components installed in the cluster you are administering, some of these services may not appear in your list of Windows services; certain data and server components may be installed on different nodes to facilitate redundancy.

Note: Sybase recommends that you only manually start and stop Sybase Unwired Platform services for debugging and troubleshooting purposes.

If you are routinely starting and stopping Unwired Server, you should use Sybase Control Center for that purpose. Sybase Control Center allows you to manage local and remote servers from a single location, and is more efficient than starting and stopping with services or desktop shortcuts.

Component	Service	Description	Dependencies
Unwired Server	Advantage Database Server	The database server that manages the messaging database.	This service must be started before Sybase Messaging Service can be started.
	OpenDS	The default LDAP server for development authentication and authorization. If you are using a provider other than LDAP, you do not need to use this service. Appears in the Developer Edition installation only.	None.

Component	Service	Description	Dependencies
	Consolidated Database (the service name depends on the data tier installation type): • If the data tier is installed with Unwired Server on the same host: SybaseUnwiredPlatform<hostname>Database<install_number> • If the data tier is installed by itself: SybaseUnwiredPlatformConsolidatedDatabase	The CDB services.	The CDB service must be started before the Unwired Server service can be started.
	SybaseUnwiredPlatform *<cluster_name>*Server *<install_number>*	The Unwired Server service.	Depends on the CDB service startup and relay server service (if used).
	Sybase Messaging Service	The synchronization service that facilitates communication with device client applications.	Depends on Advantage Database Server startup.
Sybase Control Center	Sybase Unified Agent *X.X*	Provides runtime services to manage, monitor, and control distributed Sybase resources. The agent must be running for Sybase Control Center to run.	None.

See also

- *Setting Up Data Tier Nodes* on page 16

Processes Reference

Unwired Platform Windows processes vary, depending on your components and license type.

Use this table to determine existing processes for each runtime component.

Component	Service	Processes
Unwired Server	OpenDS LDAP (Development Edition licenses only)	`OpenDS_service.exe`, `java.exe`
	Replication-based synchronization services (via MobiLink)	Consolidated database: `dbsrv11.exe` Synchronization: `mlsrv11.exe`
	Messaging-based synchronization services	`AdminWebServices.exe`, `ampservice.exe`, `JMSBridge.exe`, `LBManager.exe`, `OBMO.exe`, `OBServiceManager.exe` Messaging database: `ads.exe`
Relay server	Relay server	`rshost.exe`
	RSOE	`rsoe.exe`
Sybase Control Center	Sybase Unified Agent	`uaservice.exe`
	Sybase Control Center repository database	`dbsrv11.exe`

Security Provider Configuration Properties

Security providers implement different properties, depending on whether or not they support authentication, authorization, audit, or attribution.

Platform administrators can configure application security properties in the Sybase Control Center for Unwired Platform console. These properties are then transcribed to an XML file in the `<UnwiredPlatform_InstallDir>\Servers\UnwiredServer\Repository\CSI\` directory. A new section is created for each provider you add.

LDAP Configuration Properties

Use these properties to configure the LDAP provider used to authenticate SCC administration logins or to configure the LDAP provider used to authenticate device application logins. If you are creating a provider for device application logins, then Unwired Platform administrators use Sybase Control Center to write these properties to the `<UnwiredPlatform_InstallDir>\Servers\UnwiredServer\Repository\CSI\default.xml` file.Use these properties to configure the LDAP provider used to authenticate SCC administration logins or to configure the LDAP provider used to authenticate device application logins. If you are creating a provider for device application logins, then Unwired Platform administrators use Sybase Control Center to write these properties to the `<UnwiredPlatform_InstallDir>\Servers\UnwiredServer\Repository\CSI\default.xml` file.

Unwired Server implements a Java LDAP provider through a common security interface used by other Sybase products like Sybase Control Center.

The Java LDAP provider consists of three provider modules, each of which is in the `com.sybase.security.ldap` Java package. This is why the syntax used between Sybase Control Center provider and Unwired Server varies.

- The **LDAPLoginModule** provides authentication services. Through appropriate configuration, you can enable certificate authentication in LDAPLoginModule.
- Optional. The **LDAPAuthorizer** or **RoleCheckAuthorizer** provide authorization services for LDAPLoginModule. LDAPLoginModule works with either authorizer. In most production deployments, you must always configure your own authorizer. However, if you are authenticating against a service other than LDAP, but want to perform authorization against LDAP, you can use the LDAPAuthorizer.

  The RoleCheckAuthorizer is always used with every security configuration; however it is not displayed in Sybase Control Center.

  Only use LDAPAuthorizer when LDAPLoginModule is not used to perform authentication, but roles are still required to perform authorization checks against the LDAP data store. If you use LDAPAuthorizer, always configure properties for it explicitly. It cannot share the configuration options specified for the LDAPLoginModule (if any are configured).
- Optional. The **LDAPAttributer** provides attribution services. The attribution provider can share configuration defined for the LDAPLoginModule. That is, if no explicit configuration properties are specified for LDAPAttributer, it uses the configuration information from the LDAPLoginModule, when only one LDAPLoginModule is configured. If there are more than one, then LDAPAttibuters cannot share properties, because they would not know which LoginModule to share with; in this case you must also configure propertes for each LDAPAttributer.

Use this table to help you configure properties for one or more of the supported LDAP providers. When configuring modules or general server properties in Sybase Control Center,

note that properties and values can vary, depending on which module or server type you configure.

Property	Default Value	Description
ServerType	None	Optional. The type of LDAP server you are connecting to: • `sunone5` -- SunOne 5.x OR iPlanet 5.x • `msad2k` -- Microsoft ActiveDirectory, Windows 2000 • `nsds4` -- Netscape Directory Server 4.x • `openldap` -- OpenLDAP Directory Server 2.x The value you choose establishes default values for these other authentication properties: • RoleFilter • UserRoleMembership • RoleMemberAttributes • AuthenticationFilter • DigestMD5Authentication • UseUserAccountControl
ProviderURL	`ldap://localhost:389`	The URL used to connect to the LDAP server. Without this URL configured, Unwired Server cannot contact your server. Use the default value if the server is: • Located on the same machine as your product that is enabled with the common security infrastructure. • Configured to use the default port (389). Note that Development Editions of Unwired Platform include an OpenDS LDAP server that runs on a nonstandard port of 10389. However, most LDAP servers use the standard port of 389. Otherwise, use this syntax for setting the value: `ldap://<hostname>:<port>`

Property	Default Value	Description
DefaultSearchBase	None	The LDAP search base that is used if no other search base is specified for authentication, roles, attribution and self registration: 1. `dc=<domainname>,dc=<tld>` For example, a machine in sybase.com domain would have a search base of dc=sybase,dc=com. 2. `o=<company name>,c=<country code>` For example, this might be o=Sybase,c=us for a machine within the Sybase organization.
SecurityProtocol	None	The protocol to be used when connecting to the LDAP server. To use an encrypted protocol, use "ssl" instead "ldaps" in the url. **Note:** ActiveDirectory requires the SSL protocol when setting the value for the password attribute. This occurs when creating a user or updating the password of an existing user.
AuthenticationMethod	simple	The authentication method to use for all authentication requests into LDAP. Legal values are generally the same as those of the java.naming.security.authentication JNDI property. Choose one of: • simple — For clear-text password authentication. • DIGEST-MD5 — For more secure hashed password authentication. This method requires that the server use plain text password storage and only works with JRE 1.4 or later. See the*Java Sun* Web site for more information.

Property	Default Value	Description
AuthenticationFilter	For most LDAP servers: `(&(uid={uid})(objectclass=person))` or For Active Directory email lookups: `(&(userPrincipalName={uid})(objectclass=user))[ActiveDirectory]` For Active Directory Windows username lookups: `(&(sAMAccountName={uid})(objectclass=user))`	The filter to use when looking up the user. When performing a username based lookup, this filter is used to determine the LDAP entry that matches the supplied username. The string "{uid}" in the filter is replaced with the supplied username.
AuthenticationScope	onelevel	The authentication search scope. The supported values for this are: • `onelevel` • `subtree` If you do not specify a value or if you specify an invalid value, the default value is used.
AuthenticationSearchBase	none	The search base used to authenticate users. If this value is not specified, the LDAP DefaultSearchBase is used.

Property	Default Value	Description
BindDN	none	The user DN to bind against when building the initial LDAP connection. In many cases, this user may need read permissions on all user records. If you do not set a value, anonymous binding is used. Anonymous binding works on most servers without additional configuration. However, the LDAP attributer may also use this DN to create the users in the LDAP server. When the self-registration feature is used, this user may also need the requisite permissions to create a user record. This behavior can occur if you do not set useUserCredentialsToBind to `true`. In this case, the LDAP attributer uses this DN to update the user attributes.
BindPassword	none	BindPassword is the password for BindDN, which is used to authenticate any user. BindDN and BindPassword are used to separate the LDAP connection into units. The AuthenticationMethod property determines the bind method used for this initial connection. If you use an encrypted the password using the CSI encryption utility, append .e to the property name. For example: `CSI.loginModule.7.options.BindPassword.e=1-AAAAEgQQOLL+LpXJO8fO9T4SrQYRC9lRT1w5ePfdczQTDsP8iACk9mDAbm3F3p5a1wXWKK8+NdJuknc7w2nw5aGJlyG3xQ==`
RoleSearchBase	none	The search base used to retrieve lists of roles. If this value is not specified, the LDAP DefaultSearchBase is used.

Property	Default Value	Description
RoleFilter	For SunONE/iPlanet: `(&(objectclass=ldapsubentry) (objectclass=nsroledefinition))` For Netscape Directory Server: (`objectclass=groupofnames) (objectclass=groupofuniquenames`)) For ActiveDirectory: `(objectclass=groupofnames) (objectclass=group))`	The role search filter. This filter should, when combined with the role search base and role scope, return a complete list of roles within the LDAP server. There are several default values depending on the chosen server type. If the server type is not chosen or this property is not initialized, no roles are available.
RoleMemberAttributes	For Netscape Directory Server: member,uniquemember	The role's member attributes defines a comma-delimited list of attributes that roles may have that define a list of DN's of people who are in the role. These values are cross referenced with the active user to determine the user's role list. One example of the use of this property is when using LDAP groups as placeholders for roles. This property only has a default value when the Netscape server type is chosen.
RoleNameAttribute	cn	The attribute for retrieved roles that is the common name of the role. If this value is "dn" it is interpreted specially as the entire dn of the role as the role name.
RoleScope	onelevel	The role search scope. The supported values for this are: • `onelevel` • `subtree` If you do not specify a value or if you specify an invalid value, the default value is used.

Property	Default Value	Description
UserRoleMembershipAttributes	For iPlanet/SunONE: nsRoleDN For ActiveDirectory: memberOf For all others: none	The user's role membership attributes property is used to define an attribute that a user has that contains the DN's of all of the roles as user is a member of. These comma-delimited values are then cross-referenced with the roles retrieved in the role search base and search filter to come up with a list of user's roles.
UserFreeformRoleMembershipAttributes	None	The "freeform" role membership attribute list. Users who have attributes in this comma-delimited list are automatically granted access to roles whose names are equal to the attribute value. For example, if the value of this property is "department" and user's LDAP record has the following values for the department attribute, { "sales", "consulting" }, then the user will be granted roles whose names are "sales" and "consulting".
Referral	ignore	The behavior when a referral is encountered. The valid values are those dictated by LdapContext, for example, "follow", "ignore", "throw".
DigestMD5AuthenticationFormat	DN For OpenLDAP: username	The DIGEST-MD5 bind authentication identity format.
UseUserAccountControlAttribute	For most LDAP servers: false For ActiveDirectory: true	The UserAccountControl attribute to be used for detecting disabled user accounts, account expirations, password expirations and so on. ActiveDirectory also uses this attribute to store the above information.
controlFlag	optional	Indicates whether authentication with this login module is sufficient to allow the user to log in, or whether the user must also be authenticated with another login module. Rarely set to anything other than "sufficient" for any login module. **Note:** controlFlag is a generic login module option rather than an LDAP configuration property.

See also

- *Authentication* on page 102

- *LDAP Security Provider* on page 107
- *Active Directory Considerations* on page 108

controlFlag Attribute Values

The Sybase implementation uses the same controlFlag values and definitions as those defined in the JAAS specification.

If you stack multiple providers, you must set the controlFlag attribute for each enabled provider.

Control flag value	Description
(Default) required	The LoginModule is required to succeed. Authentication proceeds down the LoginModule list.
requisite	The LoginModule is required to succeed. Subsequent behavior depends on the authentication result: • If authentication succeeds, authentication continues down the LoginModule list. • If authentication fails, control returns immediately to the application (authentication does not proceed down the LoginModule list).
sufficient	The LoginModule is not required to succeed. Subsequent behavior depends on the authentication result: • If authentication succeeds, control returns immediately to the application (authentication does not proceed down the LoginModule list). • If authentication fails, authentication continues down the LoginModule list.
optional	The LoginModule is not required to succeed. Irrespective of success or failure, authentication proceeds down the LoginModule list.

Example

Say you list providers in the following order and set the corresponding controlFlag attributes as follows:

1. CertificateAuthenticationLoginModule (sufficient)
2. LDAP (optional)
3. NativeOS (sufficient)

A client doing certificate authentication (for example, X.509 SSO to SAP) can authenticate immediately. Subsequent modules are not called, because they are not required. If there are regular username/password credentials, then they go to LDAP (first), which may authenticate them, and set them up with roles from the LDAP groups they belong to. Then NativeOS is

invoked, and if that also succeeds, Unwired Platform picks up roles based on the Windows groups they are in.

NTProxy Configuration Properties

Configure these properties to allow the operating system's security mechanisms to validate user credentials using NTProxy (Windows Native OS). Access these properties from the Authentication tab of the Security node in Sybase Control Center.

Table 25. Authentication properties

Properties	Default Value	Description
Extract Domain From Username	true	If set to true, the user name can contain the domain in the form of *<username>@<domain>*. If set to false, the default domain (described below) is always used, and the supplied user name is sent to through SSPI untouched.
Default Domain	The domain for the host computer of the Java Virtual Machine.	Specifies the default host name, if not overridden by the a specific user name domain.
Default Authentication Server	The authentication server for the host computer of the Java Virtual Machine.	The default authentication server from which group memberships are extracted. This can be automatically determined from the local machine environment, but this property to bypass the detection step.
useFirstPass	false	If set to true, the login module attempts to retrieve only the user name and password from the shared context. It never calls the callback handler.
tryFirstPass	false	If set to true, the login module first attempts to retrieve the user name and password from the shared context before attempting the callback handler.
clearPass	false	If set to true, the login module clears the user name and password in the shared context when calling either commit or abort.

Properties	Default Value	Description
storePass	false	If set to true, the login module stores the user name and password in the shared context after successfully authenticating.

See also

- *Authentication* on page 102
- *NTProxy Security Provider* on page 111
- *Setting Up the NTProxy Provider* on page 118

Certificate Authentication Properties

Add and configure authentication provider properties for CertificateAuthenticationLoginModule, or accept the default settings.

Table 26. CertificateAuthenticationLoginModule properties

Property	Description
Implementation class	The fully qualified class that implements the login module. `com.sybase.security.core.CertificateAuthenticationLoginModule` is the default class.
Provider type	`LoginModule` is the only supported value.
Control flag	Determines how success or failure of this module affects the overall authentication decision. `optional` is the default value.
Clear password	(Optional) If true, the login module clears the user name and password from the shared context. The default is false.
Store password	(Optional) If true, the login module stores the user name and password in the shared context. The default is false.
Try first password	(Optional) If true, the login module attempts to retrieve user name and password information from the shared context, before using the callback handler. The default is false.
Use first password	(Optional) If true, the login module attempts to retrieve the user name and password only from the shared context. The default is false.
Enable revocation checking	(Optional) Enables online certificate status protocol (OCSP) certificate checking. Revoked CA certificates result in authentication failure.

Property	Description
Regex for username certificate match	(Optional) By default, this value matches that of the certificates common name (CN) property used to identify the user. If a mobile application user supplies a user name that does not match this value, authentication fails.
Trusted certificate store	(Optional) The file containing the trusted CA certificates (import the issuer certificate into this certificate store). Use this property and `Store Password` property to keep the module out of the system trust store. The default Unwired Server system trust store is `<Unwired-Platform_InstallDir\Servers\Unwired-Server\Repository\Securitytruststore\truststore.jks`. **Note:** This property is required only if Validate certificate path is set to true.
Trusted certificate store password	(Optional) The password required to access the trusted certificate store. For example, import the issuer of the certificate you are trying to authenticate into the shared JDK cacerts file and specify the password using this property. **Note:** This property is required only if Validate certificate path is set to true.
Trusted certificate store provider	(Optional) The keystore provider. For example, "SunJCE." **Note:** This property is required only if Validate certificate path is set to true.
Trusted certificate store type	(Optional) The type of certificate store. For example, "JKS." **Note:** This property is required only if Validate certificate path is set to true.

Property	Description
Validate certificate path	If true (the default), performs certificate chain validation of the certificate being authenticated, starting with the certificate being validated. Verifies that the issuer of that certificate is valid and is issued by a trusted certificate authority (CA), if not, it looks up the issuer of that certificate in turn and verifies it is valid and is issued by a trusted CA. In other words, it builds up the path to a CA that is in the trusted certificate store. If the trusted store does not contain any of the issuers in the certificate chain, then path validation fails. For example: 1. Import a CA certificate (ca.crt) that is used to issue the client certificate (sybase101.p12) into truststore.jks: `keytool -import -keystore <Unwired-Platform_installDir>\UnwiredPlatform\Servers\UnwiredServer\Repository\Security\truststore.jks -file <Unwired-Platform_installDir>\UnwiredPlatform\Servers\UnwiredServer\Repository\Security\ca.crt` 2. Set these values for these properties: • Validate certificate path = `True` • Trusted certificate store = `<UnwiredPlatform_installDir>\UnwiredPlatform\Servers\UnwiredServer\Repository\Security\truststore.jks>` • Trusted certificate store password = `changeit` • Trusted certificate store provider = `SunJCE` • Trusted certificate store type = `JCEKS`

See also

- *CertificateAuthenticationLoginModule Authentication Provider* on page 112
- *Single Sign-on for SAP* on page 123
- *Preparing Your SAP Environment for SSO* on page 128
- *Installing the SAP Cryptographic Libraries on Unwired Platform* on page 34
- *Setting up SAP SSO Using X.509 Certificates* on page 129

SAP SSO Token Authentication Properties

Add and configure authentication provider properties for SAPSSOTokenLoginModule or accept the default values

Table 27. SAPSSOTokenLoginModule properties

Property	Description
Implementation class	(Required) – the fully qualified class that implements the login module. `com.sybase.security.sap.SAPSSOTokenLoginModule` is the default class.
Provider type	(Required and read-only) – `LoginModule` is the only supported value.
Control flag	(Required) – `optional` is the default value. Determines how success or failure of this module affects the overall authentication decision.
SAP server URL	(Required) – the SAP server URL that authenticates the user and from which Unwired Server gets the SSO2 token.
Clear password	(Optional) – if set to `True`, the login module clears the username and password in the shared context.
Disable server certificate validation	(Optional) – the default is False. If set to `True`, disables certificate validation when establishing an HTTPS connection to the SAP server using the configured URL. Set to `True` only for configuration debugging.
SAP server certificate	(Optional) – name of the file containing the SAP certificate's public key in `.pse` format. This is required only when token caching is enabled by setting a SAP SSO token persistence data store value.
SAP server certificate password	(Optional) – password used to access the SAP server certificate.

Property	Description
SAP SSO token persistence data store	(Optional) – JNDI name used to look-up the data source to persist the retrieved SSO2 tokens. Set to "jdbc/default" to store tokens in the Unwired Server CDB. If unconfigured, some caching is still done based on the "Authentication cache timeout interval" property associated with the security configuration setting. If you use the default setting, you do not need to set SAP SSO token persistence data store, SAP server certificate, SAP server certificate password, or Token expiration interval properties. To enable token caching through the SAPSSOTokenLoginModule: **1.** Set the SAP SSO token persistence data store value to "jdbc/default." **2.** Download and install the SAP SSO2 token files. See *Installing the SAP SSO2Token Files on Unwired Server Hosts*. **3.** Specify the correct value for the SAP server certificate, SAP server certificate, SAP server certificate password and Token expiration interval properties.
Store password	(Optional) – if set to true, the login module stores the username/password in the shared context after successfully authenticating the user.
Token expiration interval	(Optional) – this property is ignored when the SAP SSO token persistence data store property is not configured. It specifies the token validity period, after which time a new token is retrieved from the SAP EIS. The default value is 120 seconds. Keep in mind that: • The "Token expiration interval" cannot exceed the "Token validity period", which is the amount of time defined in the back-end SAP server for which the token is valid. • The "Authentication cache timeout" property must be less than the "Token expiration interval" property value.

Property	Description
Try first password	(Optional) – if set to `True`, the login module attempts to retrieve the username/password from the shared context, before calling the callback handler.
Use first password	(Optional) – if set to `True`, the login module attempts to retrieve the username/password only from the shared context, and never calls the callback handler.

See also

- *SAPSSOTokenLoginModule Authentication Provider* on page 111
- *Preparing Your SAP Environment for SSO* on page 128
- *Installing the SAP SSO2Token Files on Unwired Server Hosts* on page 135

EIS Data Source Connection Properties Reference

Name and configure connection properties when you create connection pools in Sybase Control Center to enterprise information systems (EIS) .

See also

- *EIS Connections* on page 30
- *EIS Connection Management Overview* on page 162

JDBC Properties

Configure Java Database Connectivity (JDBC) connection properties.

This list of properties can be used by all datasource types. Sybase does not document native properties used only by a single driver. However, you can also use native driver properties, naming them using this syntax:

```
<driver_type>:<NativeConnPropName>=<SupportedValue>
```

Note: If Unwired Server is connecting to a database with a JDBC driver, ensure you have copied required JAR files to correct locations. See *Preparing Unwired Server to Connect to JDBC Databases*.

Name	Description	Supported values
After Insert	Changes the value to `into` if a database requires `insert into` rather than the abbreviated `into`.	`into`

Name	Description	Supported values
Batch Delimiter	Sets a delimiter, for example, a semicolon, that can be used to separate multiple SQL statements within a statement batch.	`<delimiter>`
Blob Updater	Specifies the name of a class that can be used to update database BLOB (long binary) objects when the BLOB size is greater than psMaximumBlobLength.	`<class name>` The class must implement the `com.sybase.djc.sql.BlobUpdater` interface.
Clob Updater	Specifies the name of a class that can be used to update database CLOB (long string) objects when the CLOB size is greater than psMaximumClobLength.	`<class name>` The class must implement the `com.sybase.djc.sql.ClobUpdater` interface.
Code Set	Specifies how to represent a repertoire of characters by setting the value of CS_SYB_CHARSET for this datasource. Used when the data in the datasource is localized. If you do not specify the correct code set, characters may be rendered incorrectly.	`[server]` If the value is server, the value of the current application server's defaultCodeSet property is used.

Name	Description	Supported values
Commit Protocol	Specifies how Unwired Server handles connections for a datasource at commit time, specifically when a single transaction requires data from multiple endpoints. If you use XA, the recovery log is stored in the tx_manager datasource, and its commit protocol must be optimistic. If tx_manager is aliased to another datasource (that is, one that is defined with the aliasFor property), the commit protocol for that datasource must be optimistic. A last-resource optimization ensures full conformance with the XA specification. The commit protocol for all other datasources should be XA_2PC. Alternately, a transaction that accesses multiple datasources for which the commit protocols are optimistic is permitted.	`[optimistic| pessimistic | XA_2PC]` Choose only one of these protocols: • Optimistic – enables connections to be committed without regard for other connections enlisted in the transaction, assuming that the transaction is not marked for rollback and will successfully commit on all resources. Note: if a transaction accesses multiple data sources with commit protocol of "optimistic", atomicity is not guaranteed. • Pessimistic – specifies that you do not expect any multi-resource transactions. An exception will be thrown (and transaction rolled back) if any attempt is made to use more than one "pessimistic" data source in the same transaction. • XA_2PC – specifies use of the XA two phase commit protocol. If you are using two phase commit, then the recovery log is stored in the "tx_manager" data source, and that data source (or the one it is aliased to) must have the commit protocol of "optimistic" or "pessimistic". All other data sources for which atomicity must be ensured should have the "XA_2PC" commit protocol.

Name	Description	Supported values
Datasource Class	Sets the class that implements the JDBC datasource. Use this property (along with the driverClass property) only if you do not have a predefined database-type entry in Unwired Server for the kind of SQL database you are connecting to. For example, you must use this property for MySQL database connections. You can implement a datasource class to work with a distributed transaction environment. Because Unwired Server supports distributed transactions, some datasources may require that a datasource class be implemented for Unwired Server to interact with it. For two-phase transactions, use the xaDataSourceClass connection property instead.	`<com.`*`mydata-source`*`.jdbc.Driver>`
Database Command Echo	Echoes a database command to both the console window and the server log file. Use this property to immediately see and record the status or outcome of database commands. When you enable this property, Unwired Server echoes every SQL query to `ml.log`, which may help you debug your application.	`[true|false]` Set a value of 1 to echo the database commands like `databaseStartCommand`, and `databaseStopCommand`. Otherwise, do not set this property, or use a value of 0 to disable the echo.

Name	Description	Supported values
Database Create Command	Specifies the operating system command used to create the database for this datasource. If this command is defined and the file referenced by ${databaseFile} does not exist, the command is run to create the database when an application component attempts to obtain the first connection from the connection pool for this datasource.	`<command>` Example: `<UnwiredPlatform_InstallDir>\Servers\SQLAnywhere11\BIN32\dbinit -q ${databaseFile}`
Database File	Indicates the database file to load when connecting to a datasource. Use this property when the path to the database file differs from the one normally used by the database server. If the database you want to connect to is already running, use the databaseName connection parameter.	`<string>` Supply a complete path and file name. The database file you specify must be on the same host as the server.

Name	Description	Supported values
Database Name	Identifies a loaded database with which to establish a connection, when connecting to a datasource. Set a database name, so you can refer to the database by name in other property definitions for a datasource. If the database to connect to is not already running, use the databaseFile connection parameter so the database can be started. **Note:** For Unwired Server, you typically do not need to use this property. Usually, when you start a database on a server, the database is assigned a name. The mechanism by which this occurs varies. An administrator can use the DBN option to set a unique name, or the server may use the base of the file name with the extension and path removed.	`[DBN|default]` If you set this property to default, the name is obtained from the DBN option set by the database administrator. If no value is used, the database name is inherited from the database type.
Database Start Command	Specifies the operating system command used to start the database for this datasource. If this command is defined and the database is not running, the command is run to start the database when the datasource is activated.	`<command>` Example: `<UnwiredPlatform_InstallDir>\Servers\SQLAnywhere11\BIN32\dbsrv11.exe`

Database Stop Command	Specifies the operating system command used to stop the database for this datasource. If this property is defined and the database is running, this command executes during shutdown.	`<command>`
		
		For a Adaptive Server™ Anywhere database, where the user name and password are the defaults (dba and sql), enter:
		
		`<UnwiredPlatform_InstallDir>\Servers\SQLAnywhere11\BIN32\dbsrv11.exe`

Name	Description	Supported values
Database Type	Specifies the database type.	`<database type>`
Database URL	Sets the JDBC URL for connecting to the database if the datasource requires an Internet connection. Typically, the server attempts to construct the database URL from the various connection properties you specify (for example, portNumber, databaseName). However, because some drivers require a special or unique URL syntax, this property allows you to override the server defaults and instead provide explicit values for this URL.	`<JDBCurl>` The database URL is JDBC driver vendor-specific. For details, refer to the driver vendor's JDBC documentation.
Driver Class	Sets the name of the class that implements the JDBC driver. Use this property (along with the dataSourceClass property) only if you do not have a predefined database-type entry in Unwired Server for the kind of SQL database you are connecting to. For example, MySQL database connections require you to use this connection property. To create a connection to a database system, you must use the compatible JDBC driver classes. Sybase does not provide these classes; you must obtain them from the database manufacturer.	*`<Class.forName("foo.bar.Driver")>`* Replace <Class.forName("foo.bar.Driver")> with the name of your driver.
Driver Debug	Enables debugging for the driver.	`[true|false]` Set to true to enable debugging, or false to disable.
Driver Debug Settings	Configures debug settings for the driver debugger.	`[default|<setting>]` The default is STATIC:ALL.

Name	Description	Supported values
Initial Pool Size	Sets the initial number of connections in the pool for a datasource. In general, holding a connection causes a less dramatic performance impact than creating a new connection. Keep your pool size large enough for the number of concurrent requests you have; ideally, your connection pool size should ensure that you never run out of available connections. The initialPoolSize value is applied to the next time you start Unwired Server.	`<int>` Replace <int> with an integer to preallocate and open the specified number of connections at start-up. The default is 0. Sybase suggests that you start with 0, and create additional connections as necessary. The value you choose allows you to create additional connections before client synchronization requires the server to create them.
Is Download Zipped	Specifies whether the driver file downloaded from jdbcDriverDownloadURL is in `.ZIP` format. This property is ignored if the value of jdbcDriverDownloadURL connection is an empty string.	`[True|False]` The default is false. The file is copied, but not zipped to `<UnwiredPlatform-install>\lib\jdbc`. Set isDownloadZipped to true to save the file to `<UnwiredPlatform-install>\lib\jdbc` and unzip the archived copy.
JDBC Driver Download URL	Specifies the URL from which you can download a database driver. Use this property with isDownloadZipped to put the driver in an archive file before the download starts.	`<URL>` Replace <URL> with the URL from which the driver can be downloaded.

Name	Description	Supported values
Language	For those interfaces that support localization, this property specifies the language to use when connecting to your target database. When you specify a value for this property, Unwired Server: • Allocates a CS_LOCALE structure for this connection • Sets the CS_SYB_LANG value to the language you specify • Sets the Microsoft SQL Server CS_LOC_PROP connection property with the new locale information Unwired Server can access Unicode data in an Adaptive Server® 12.5 or later, or in Unicode columns in Adaptive Server 12.5 or later. Unwired Server automatically converts between double-byte character set (DBCS) data and Unicode, provided that the Language and CodeSet parameters are set with DBCS values.	`<language>` Replace <language> with the language being used.
Max Idle Time	Specifies the number of seconds an idle connection remains in the pool before it is dropped.	`<int>` If the value is 0, idle connections remain in the pool until the server shuts down. The default is 60.

Name	Description	Supported values
Max Pool Size	Sets the maximum number of connections allocated to the pool for this datasource. Increase the maxPoolSize property value when you have a large user base. To determine whether a value is high enough, look for ResourceMonitorTimeoutException exceptions in `<hostname>-server.log`. Continue increasing the value, until this exception no longer occurs. To further reduce the likelihood of deadlocks, configure a higher value for maxWaitTime. To control the range of the pool size, use this property with minPoolSize.	`<int>` A value of 0 sets no limit to the maximum connection pool size.
Max Wait Time	Sets the maximum number of seconds to wait for a connection before the request is cancelled.	`<int>` The default is 60.
Max Statements	Specifies the maximum number of JDBC prepared statements that can be cached for each connection by the JDBC driver. The value of this property is specific to each JDBC driver.	`<int>` A value of 0 (default) sets no limit to the maximum statements.
Min Pool Size	Sets the minimum number of connections allocated to the pool for this datasource.	`<int>` A value of 0 (default) sets no limit to the minimum connection pool size.

Name	Description	Supported values
Network Protocol	Sets the protocol used for network communication with the datasource. Use this property (along with the driverClass, and dataSourceClass properties) only if you do not have a predefined database-type entry in Unwired Server for the kind of SQL database you are connecting to. For example, you may be required to use this property for MySQL database connections.	The network protocol is JDBC driver vendor-specific. There are no predefined values. See the driver vendor's JDBC documentation.
Password	Specifies the password for connecting to the database.	`[default|<password>]`
Ping and Set Session Auth	Runs the ping and session-authorization commands in a single command batch; may improve performance. You can only enable the Ping and Set Session Auth property if you have enabled the Set Session Auth property so database work runs under the effective user ID of the client.	`[True|False]` Set to true to enable, or false to disable.
Ping Connections	Pings connections before attempting to reuse them from the connection pool.	`[True|False]` Set to true to enable ping connections, or false to disable.
Ping SQL	Specify the SQL statement to use when testing the database connection with ping.	`[default|<statement>]` Replace <statement> with the SQL statement identifier. The default is "select 1".
Port Number	Sets the server port number where the database server listens for connection requests.	`[default|<port>]` Replace <port> with the TCP/IP port number to use (that is, 1 – 65535). If you set the value as default, the default protocol of the datasource is used.

Name	Description	Supported values
PS Maximum Blob Length	Indicates the maximum number of bytes allowed when updating a BLOB datatype using PreparedStatement.setBytes.	`[default|<int>]` Replace <int> with the number of bytes allowed during an update. The default is 16384.
PS Maximum Clob Length	Indicates the maximum number of characters allowed when updating a CLOB datatype using PreparedStatement.setString.	`[default|<int>]` Replace <int> with the number of bytes allowed during an update. The default is 16384.
Role Name	Sets the database role that the user must have to log in to the database.	`[default|<name>]` If you set this value to default, the default database role name of the datasource is used.
Server Name	Defines the host where the database server is running.	`<name>` Replace <name> with an appropriate name for the server.
Service Name	Defines the service name for the datasource. For SQL Anywhere servers, use this property to specify the database you are attaching to.	`<name>` Replace <name> with an appropriate name for the service.
Set Session Auth	Establishes an effective database identity that matches the current mobile application user. If you use this property, you must also use setSessionAuthSystemID to set the session ID. Alternately you can pingAndSetSessionAuth if you are using this property with pingConnection. The pingAndSetSessionAuth property runs the ping and session-authorization commands in a single command batch, which may improve performance.	`[true|false]` Choose a value of 1 to use an ANSI SQL set session authorization command at the start of each database transaction. Set to 0 to use session-based authorizations.

Name	Description	Supported values
Set Session Auth System ID	If Set Session Authorization is enabled, specifies the database identity to use when the application server accesses the database from a transaction that runs with "system" identity.	`<database identity>` Replace <database identity> with the database identifier.
Start Wait	Sets the wait time (in seconds) before a connection problem is reported. If the start command completes successfully within this time period, no exceptions are reported in the server log. startWait time is used only with the databaseStartCommand property.	`<int>` Replace <int> with the number of seconds Unwired Server waits before reporting an error.
Truncate Nanoseconds	Sets a divisor/multiplier that is used to round the nanoseconds value in a java.sql.Timestamp to a granularity that the DBMS supports.	`[default|<int>]` The default is 10 000 000.
Use Quoted Identifiers	Specifies whether or not SQL identifiers are quoted.	`[True|False]` Set to true to enable use of quoted identifiers, or false to disable.
User	Identifies the user who is connecting to the database.	`[default|<user name>]` Replace <user name> with the database user name.
XA Datasource Class	Specifies the class name or library name used to support two-phase commit transactions, and the name of the XA resource library.	`<class name>` Replace <class name> with the class or library name. • SQL Anywhere database: `com.sybase.jdbc3.jdbc.SybXADataSource` • Oracle database: `oracle.jdbc.xa.client.OracleXADataSource`

SAP Java Connector Properties

Configure SAP Java Connector (JCo) connection properties.

For a comprehensive list of SAP JCo properties you can use to create an instance of a client connection to a remote SAP system, see *http://help.sap.com/javadocs/NW04/current/jc/com/sap/mw/jco/JCO.html#createClient(java.util.Properties)*.

Note: If Unwired Server is connecting to SAP with a Java connector, ensure you have copied required files to correct locations.

Table 28. General connection parameters

Name	Description	Supported values
Client Number	Specifies the SAP client.	Three-digit client number; preserve leading zeros if they appear in the number
Logon User	Specifies the login user ID.	User name for logging in to the SAP system If using X.509 certificate authentication, remove the JCo properties `jco.client.passwd` and `jco.client.user` defined for the SAP connection profile in Sybase Control Center (SCC).
Password	Specifies the login password.	Password for logging in to the SAP system
Language	Specifies a login language.	ISO two-character language code (for example, EN, DE, FR), or SAP-specific single-character language code. As a result, only the first two characters are ever used, even if a longer string is entered. The default is EN.
System Number	Indicates the SAP system number.	SAP system number
Host Name	Identifies the SAP application server.	Host name of a specific SAP application server
Message Server	Identifies the SAP message server.	Host name of the message server
Gateway Host	Identifies the SAP gateway host.	Host name of the SAP gateway Example: `GWHOST=hs0311`

Name	Description	Supported values
Gateway Service	Identifies the SAP gateway service.	Service name of the SAP gateway Example: `GWSERV=sapgw53`
R/3 Name	Specifies R/3 name.	Name of the SAP system
Server Group	Identifies the group of SAP application servers.	Group name of the application servers
External Server Program	Identifies the program ID of the external server program.	Path and name of the external RFC server program, or program ID of a registered RFC server program Example: `TPNAME=/sap/srfcserv`
External Server Program Host	Identifies the host of the external server program. This information determines whether the RFC client connects to an RFC server started by the SAP gateway or to an already registered RFC server. **Note:** If the gateway host and external server program host are different, make sure that the SAP gateway has access to start the server program through a remote shell.	Host name of the external RFC server program Example: `TPHOST=hs0311`
Remote Host Type	Identifies the type of remote host.	2: R/2 3: R/3 E: external
RFC Trace	Specifies whether or not to enable RFC trace.	0: disable 1: enable
Initial Codepage	Identifies the initial code page in SAP notation. A code page is used whenever character data is processed on the application server, appears on the front end, or is rendered by a printer.	Four-digit SAP code page number

Name	Description	Supported values
Enable ABAP Debugging	Enables or disables ABAP debugging. If enabled, the connection is opened in debug mode and the invoked function module can be stepped through in the debugger. For debugging, an SAP graphical user interface (SAPGUI) must be installed on the same machine the client program is running on. This can be either a normal Windows SAPGUI or a Java GUI on Linux/UNIX systems.	0: no debugging 1: attach a visible SAPGUI and break at the first ABAP statement of the invoked function module
Remote GUI	Specifies whether a remote SAP graphical user interface (SAPGUI) should be attached to the connection. Some older BAPIs need an SAPGUI because they try to send screen output to the client while executing.	0: no SAPGUI 1: attach an "invisible" SAPGUI, which receives and ignores the screen output 2: attach a visible SAPGUI For values other than 0 a SAPGUI needs to be installed on the machine, where the client program is running. This can be either a Windows SAPGUI or a Java GUI on Linux/Unix systems.
Get SSO Ticket	Generates an SSO2 ticket for the user after login to allow single sign-on. If RfcOpenConnection() succeeds, you can retrieve the ticket with RfcGetPartnerSSOTicket() and use it for additional logins to systems supporting the same user base.	0: do not generate SSO2 ticket 1: generate SSO2 ticket
Use Cookie Version 2	Indicates whether or not to use the specified SAP Cookie Version 2 as the login ticket instead of user ID and password.	User: $MYSAPSSO2$ Password: Base64-encoded ticket Login with single sign-on is based on secure network connection (SNC) encryption and can only be used in combination with an SNC.

Name	Description	Supported values
Use X509	Indicates whether or not to use the specified X509 certificate as the login certificate instead of user ID and password.	User: $X509CERT$ Password: Base64-encoded ticket Login with X509 is based on secure network connection (SNC) encryption and can only be used in combination with an SNC.
Logon Check	Enables or disables login check at open time.	0: disable 1: enable If you set this to 0, RfcOpenConnection() opens a network connection, but does not perform the login procedure. Therefore, no user session is created inside the back-end system. This parameter is intended only for executing the function module RFC_PING.
Additional GUI Data	Provides additional data for graphical user interface (GUI) to specify the SAProuter connection data for the SAPGUI when it is used with RFC.	/H/ *router string* : the entire router string for the SAPGUI /P/ *password* : specify this value if the password for the SAPGUI connection is not the same as the password for the RFC connection.
GUI Redirect Host	Identifies which host to redirect the remote graphical user interface to.	Host name
GUI Redirect Service	Identifies which service to redirect the remote graphical user interface to.	Name of the service
Remote GUI Start Program	Indicates the program ID of the server that starts the remote graphical user interface.	Program ID of the server
SNC Mode	Enables or disables secure network connection mode.	0: off 1: on
SNC Partner	Identifies the secure network connection partner.	Secure network connection name of the application server (for example, p:CN=R3, O=XYZ-INC, C=EN)

Name	Description	Supported values
SNC Level	Specifies the secure network connection security level.	1: digital signature 2: digital signature and encryption 3: digital signature, encryption, and user authentication 8: default value defined by backend system 9: maximum value that the current security product supports
SNC Name	Indicates the secure network connection name. This property overrides the default secure network connection partner.	Token or identifier representing the external RFC program
SNC Service Lib Path	Identifies the path to the library that provides secure network connection service.	Full path and name of third-party security library
R/2 Destination	Identifies a configured R/2 system defined in the sideinfo configuration.	
Logon ID	Defines the string for SAPLOGON on 32-bit Windows.	String key to read parameters from the saplogon.ini file created by the SAPLogon GUI program on Windows
External Authentication Data	Provides data for external authentication (PAS). This is an old login mechanism similar to SSO; Sybase recommends that you do not use this approach.	
External Authentication	Specifies type of external authentication (PAS). See External Authentication Data property.	

SAP DOE-C Properties

Configure SAP Data Orchestration Engine Connector (DOE-C) properties. This type of connection is available in the list of connection templates only when you deploy a DOE-C package. No template exists for these types of connections.

Note: If you change the username or password property of a DOE-C connection, you must reopen the same dialog and click `Test Connection` after saving. Otherwise the error state

of this DOE-C package is not set properly, and an error message is displayed. This will not work if you click `Test Connection` before saving the properties.

Name	Description	Supported values
Username	Specifies the SAP user account ID. The SAP user account is used during interaction between the connected SAP system and client for certain administrative activities, such as sending acknowledgment messages during day-to-day operations or "unsubscribe" messages if a subscription for this connection is removed. This account is not used for messages containing business data; those types of messages are always sent within the context of a session authenticated with credentials provided by the mobile client. The technical user name and password must be set to perform actions on subscriptions.	Valid SAP login name for the DOE host system.
Password	Specifies the password for the SAP user account.	Valid password.
DOE SOAP Timeout	Specifies a timeout window during which unresponsive DOE requests are aborted.	Positive value (in seconds). The default is 420 (7 minutes).
DOE Extract Window	Specifies the number of messages allowed in the DOE extract window.	Positive value (in messages). The default is 50. When the number of messages in the DOE extract window reaches 50% of this value, DOE-C sends a `StatusReqFromClient` message, to advise the SAP DOE system of the client's messaging status and acknowledge the server's state. The default value is 50.

Name	Description	Supported values
Packet Drop Size	Specifies the size, in bytes, of the largest JavaScript Object Notation (JSON) message that the DOE connector processes on behalf of a JSON client. The packet drop threshold size should be carefully chosen, so that it is larger than the largest message sent from the DOE to the client, but smaller than the maximum message size which may be processed by the client. Messages larger than the packet drop threshold size causes the subscription to enter the DOE packet drop state and become unusable.	Positive value (in bytes). The default is 1MB. Do not set higher than 2MB, or lower than 4096.
Service Address	Specifies the DOE URL.	Valid DOE URL. If you are using DOE-C with SSO: • Modify the port from the standard http://host:8000 to https://host:8001/. • Add the certificate being used as the technical user and DOE-C endpoint security profile certificate to the SAP DOE system's SSL Server certificate list by using the STRUST transaction. See your SAP documentation for details.
Listener URL	Specifies the DOE-C server listener URL.	Valid DOE-C listener URL.
Security Profile	Specifies the security profile for the DOE-C endpoint.	Valid security profile.

Name	Description	Supported values
SAP Technical User Certificate Alias	Sets the alias for the Unwired Platform keystore entry that contains the X.509 certificate for Unwired Server's SSL peer identity. If you do not set a value, mutual authentication for SSL is not used when connecting to the Web service. If you are using DOE-C with SSO use the "SAP Technical User Certificate Alias" only for configurations which require the technical user to identify itself using an X.509 certificate; it specifies the Certificate Alias to be used as the technical user. This overrides the "Username" and "Password" settings normally used.	Valid certificate alias.

Web Services Properties

Configure connection properties for the Simple Object Access Protocol (SOAP) and Representational State Transfer (REST) architectures.

Name	Description	Supported values
Password	Specifies the password for HTTP basic authentication, if applicable.	Password
Address	Specifies a different URL than the port address indicated in the WSDL document at design time.	HTTP URL address of the Web service
User	Specifies the user name for HTTP basic authentication, if applicable.	User name
Certificate Alias	Sets the alias for the Unwired Platform keystore entry that contains the X.509 certificate for Unwired Server's SSL peer identity. If you do not set a value, mutual authentication for SSL is not used when connecting to the Web service.	Use the alias of a certificate stored in the Unwired Server certificate store.

Command Line Utilities

Invoke command line utilities directly from the operating system.

Unwired Platform includes a set of command line utilities for carrying out routine administrative tasks, such as deploying a package or maintaining a consolidated database (CDB). You can also include these commands in batch files for repeated use.

To launch a utility:

- Double-click its icon, if it has one.
- If it does not have an icon in the program group, enter the utility program command at the Windows command prompt.

Relay Server Utilities

Use relay server utilities to administer the relay server: rshost and regRelayServer.

Launch these utilities from the command line; they are not available from any other administration tool.

See also

- *Chapter 5, Relay Server Clusters* on page 53
- *Relay Server Documentation* on page 54
- *Relay Server Configuration Files* on page 363
- *Configuring RSOE Logging* on page 258

Relay Server Host (rshost) Utility

Maintains relay server state information across client requests and RSOE sessions and manages the log file used by the relay server. This utility is located in *UnwiredPlatform_InstallDir*`\Servers\SQLAnywhere11\Mobilink\relayserver\IIS\Bin32\IAS_relay_server\Server`.

Syntax

Variable declaration:

```
rshost [option]+
```

Parameters

- <***option***> – the following options can be used to configure `rshost`. They are all optional.

Option	Description
`-f <filename>`	File name of the relay server configuration file.
`-o <filename>`	File name to use for logging.
`-oq`	Prevent popup window on startup error.
`-q`	Run in minimized window.
`-qc`	Close window on completion.
`-u`	Update configuration of a running relay server.
`-ua`	Archive the log file to <yymmdd><nn>.log and truncate.

Examples

- **Example 1: Updating the relay server configuration –** The following command updates rshost with changes made to relay server configuration:

  ```
  rshost –u –f rs.config
  ```

- **Example 2: Setting configuration and log files for rshost –** The following command specifies the configuration and log files for the relay server host; it configures the `rshost` to run in a minimized window and close the window on completion:

  ```
  rshost -q -qc -f C:\Sybase\UnwiredPlatform\Servers
  \SQLAnywhere11\MobiLink\relayserver\IIS
  \Bin32\IAS_relay_server\Server\rs.config -o c:\Sybase\log
  \rs.log
  ```

  For more information on the relay server host utility, see *rshost utility* in the SQL Anywhere documentation.

See also

- *Register Relay Server (regRelayServer) Utility* on page 330

Register Relay Server (regRelayServer) Utility

Use regRelayServer.bat to perform multiple relay server and RSOE administrative actions. The same actions you perform with this utility can also be performed in Sybase Control Center.

This command is located in the `<UnwiredPlatform_InstallDir>\Servers\UnwiredServer\bin\regRelayServer.bat` directory.

Syntax

```
regRelayServer [config | export | exportRSconfig | migrate|
quickconfig | remove | start | stop]
```

Parameters

- **config –** Configure relay server or an RSOE for an Unwired Server node with a named XML file as input. Use this command syntax:

  ```
  regRelayServer.bat config -f <filename> -sl
  ```

  Options:

  - **-f <*filename*> –** The input file and path.
  - **-sl –** Disable all user interactive questions (run silently).
- **export –** Export the existing RSOE configuration into a file. Use this command syntax:

  ```
  regRelayServer.bat export -o <filename> -sl
  ```

  Options:

  - **-o <*filename*> –** The output file and path.
  - **-sl –** Disable all user interactive questions (run silently).
- **exportRSconfig –** Export the actual relay server configuration properties file, which could be used by the Unwired Platform administrator to configure Unwired Server farms, server nodes, for remaining relay servers that require configuration or updates. Use this command syntax:

  ```
  regRelayServer.bat exportRSconfig -host -o <filename> -p <port> -
  sl
  ```

  Options:

  - **-host –** either the relay server host name or the host name of the load balancer (if one is used in your environment). Device clients use the host name to establish connections. Other relay servers can use the host name to identify peers if a load balancer is not used.
  - **-o <*filename*> –** The output file and path.
  - **-p <*port*> –** the HTTP port for relay server connections.
  - **-sl –** Disable all user interactive questions (run silently).
- **migrate –** Migrate the configuration values defined in the 1.5.5 version of `<UnwiredPlatfrom_InstallDir>\Servers\UnwiredServer\config\relayserver.properties` file and port them to the current version of the cluster database. Use this command syntax:

  ```
  regRelayServer.bat migrate -host -o <filename> -sl
  ```

  Options:

  - **-o <*filename*> –** Store the command output into the named file to ensure that the migration is successful.
  - **-sl –** Disable all user interactive questions (run silently).
- **quickconfig –** Rapidly create a relay server configuration, collecting only connection property values and use system-generate defaults for the remaining properties required. RSOEs deployed with this method require manual property edits before they can be

started. Export the configuration and update the properties in the file. You can then re-apply the configuration with the configure command. Use this command syntax:

```
regRelayServer.bat quickconfig -d <description> -host <name> -p
<port> -s <URLsuffix> -sl -sp <port>
```

- **-d <*description*> –** description for the relay server.
- **-host –** either the relay server host name or the host name of the load balancer (if one is used in your environment). Device clients use the host name to establish connections. Other relay servers can use the host name to identify peers if a load balancer is not used.
- **-p <*port*> –** the HTTP port for relay server connections.
- **-s <*URLsuffix*> –** A URL suffix that allows a device application client to connect to a relay server farm. The default value for Apache on Linux is `/srv/iarelayserver`, and the default value for IIS on Windows is `/ias_relay_server/server/rs_server.dll`.

  For example, to connect a client to a relay server farm called RBSFarm1, you might use `/sup_relay_server/client/rs_client.dll/RBSFarm1` as your URL suffix.
- **-sl –** Disable all user interactive questions (run silently).
- **-sp <*port*> –** the secure HTTP port for relay server connections.

- **remove –** Deletes one or all RSOE process for the Unwired Server node. However, the configuration file is not deleted, because other RSOEs on the server node may require it. Supported options include:

  - **-f <*name*> –** The relay server farm name.
  - **-host –** either the relay server host name or the host name of the load balancer (if one is used in your environment). Device clients use the host name to establish connections. Other relay servers can use the host name to identify peers if a load balancer is not used.
  - **-p <*port*> –** the HTTP port for relay server connections.

  If you do not specify -h, -p, and -f options, all RSOEs for the Unwired Server node are removed. Otherwise, only those identified are removed.
- **start –** Start one or all RSOE processes for the Unwired Server node. Use this command syntax:

  ```
  regRelayServer.bat start -d <time> -f <name> -host <name> -os
  <size> -ot -p <port> -v <int>
  ```

  Options:

  - **-d <*time*> –** When a connection fails, how long, in seconds to wait before retrying. The default is 5 seconds.
  - **-f <*name*> –**
  - **-host –** either the relay server host name or the host name of the load balancer (if one is used in your environment). Device clients use the host name to establish connections.

Other relay servers can use the host name to identify peers if a load balancer is not used.

- **-os –** The maximum log file output size. The default is 10240 KB.
- **-ot –** Truncates the log file after reaching the size specified by the maximum output size.
- **-p <*port*> –** the HTTP port for relay server connections.
- **-v <*int*> –** The output log verbosity level. Set any value in the range of 0-5. For troubleshooting and maintenance, levels 1 or 2 should be sufficient.

- **stop –** Stop one or all RSOE processes for the Unwired Server node. Supported options include:
  - **-f <*name*> –** the server farm for which the relay server manages requests. This property is case-sensitive. The configured value must match the value defined for the RSOE or the connection fails.
  - **-host –** either the relay server host name or the host name of the load balancer (if one is used in your environment). Device clients use the host name to establish connections. Other relay servers can use the host name to identify peers if a load balancer is not used.
  - **-p <*port*> –** the HTTP port for relay server connections.

Examples

- **Export properties then configure a new relay server –** In this example, an administrator uses this utility multiple times but on different host computers:
  1. The administrator exports the file for a relay server deployed in a test environment. This configuration is stable and the administrator now wishes to propagate these properties to all new relay servers in a production environment. The administrator runs this command on the test host computer:
     ```
     regRelayServer.bat exportRSconfig -o <path>\rsconfig.xml
     ```
  2. The administrator transfers the file to each new host. The command used to apply the file on the new host is:
     ```
     regRelayServer.bat config -f <path>\rsconfig.xml
     ```
- **Registering a new RSOE and configuring it with rsoe.config.template.xml –** In this example an administrator has copied and modified `<UnwiredPlatfrom_InstallDir>\Servers\UnwiredServer\config\rsoeConfig.template.xml.`. Because the XML elements do not include an `ID` attribute value, this utility will treat the RSOE as newly deployed, and register it in the cluster database in addition to configuring it. The administrator uses this command:
  ```
  regRelayServer.bat config -f <path>\rsoe.config.template.xml
  ```
- **Start all RSOEs on the Unwired Server node –** Once RSOEs are configured, rather than restarting the host computer, the administrator starts all newly configured RSOEs on the current machine with this command:

```
regRelayServer.bat start
```

- **Stop only RSOEs of a named relay server farm –** An administrator needs to make a change to one of the RSOEs recently started, and uses this command:

```
regRelayServer.bat stop -f myhost.MBS.farm -h myUServerhost -p
5001
```

Usage

- For all commands: ensure the Unwired Server node cluster database is running before you use this utility.
- For start and stop commands: if you do not specify -host, -p, and -f options, all RSOEs for the Unwired Server node start. Otherwise, only those identified are started.
- For the migrate command:
  - The `relayserver.properties` file must exist and must use supported values for the relay server and RSOE. Ensure that no changes to the file occur while the file is being processed by this command.
  - This migrate command is only available until `relayserver.properties` has been migrated for a specific relay server host.
  - Once you migrate `relayserver.properties`, do not update it.

See also

- *Setting and Distributing Properties for Multiple Relay Servers* on page 68
- *Relay Server Host (rshost) Utility* on page 329

Certificate and Key Management Utilities

Use the certificate management utilities to encrypt Unwired Server ports.

Launch these utilities from the command line; the certificate and key management utilities are not available from any other administration tool.

Certificate Creation (createcert) Utility

Generates X.509 certificates or signs pregenerated certificate requests. This utility is located in `<UnwiredPlatform_InstallDir>\Servers\SQLAnywhere11\BIN32`.

You may choose to purchase certificates from a third party. These certificate authorities (CAs) provide their own tools for creating certificates. You can use **createcert** to create certificates for development and testing; you can also use it for production certificates.

Syntax

```
createcert [options]
```

Parameters

- **[*options*] –** these options are available through the **createcert** utility:

Option	Description
`-r`	Creates a PKCS #10 certificate request. **createcert** does not prompt for a signer or any other information used to sign a certificate.
`-s <filename>`	Signs the PKCS #10 certificate request that is in the specified file. The request can be DER or PEM encoded. **createcert** does not prompt for key generation or subject information.

Note: To create a signed certificate, use **createcert** without options. To break the process into two separate steps, for example so one person creates a request and another person signs it, the first person can run **createcert** with `-r` to create a request and the second person can sign the request by running **createcert** with `-s`.

When you run createcert, you are asked for all or some of this information, depending on whether you specified `-r` or `-s` or neither.

- `Choose encryption type` – select RSA.
- `Enter RSA key length (512-16384)` – this prompt appears only if you chose RSA encryption. Specify a length between 512 bits and 16384 bits.
- Subject information – enter this information, which identifies the entity:
  - `Country Code`
  - `State/Province`
  - `Locality`
  - `Organization`
  - `Organizational Unit`
  - `Common Name`
- `Enter file path of signer's certificate` – (optional) supply a location and file name for the signer's certificate. If you supply this information, the generated certificate is a signed certificate. If you do not supply this information, the generated certificate is a self-signed root certificate.
- `Enter file path of signer's private key` – supply a location and file name to save the private key associated with the certificate request. This prompt appears only if you supplied a file in the previous prompt.
- `Enter password for signer's private key` – supply the password that was used to encrypt the signer's private key. Supply this password only if the private key was encrypted.
- `Serial number` – (optional) supply a serial number. The serial number must be a hexadecimal string of 40 digits or less. This number must be unique among all

certificates signed by the current signer. If you do not supply a serial number, createcert generates a GUID as the serial number.

- `Certificate will be valid for how many years (1-100)` – specify the number of years that the certificate is valid. After this period, the certificate expires, along with all certificates it signs.
- `Certificate Authority (y)es or (n)o` – indicate whether this certificate can be used to sign other certificates. By default, certificates are not certificate authorities (n).
- `Key usage` – supply a comma-separated list of numbers that indicate how the certificate's private key can be used. The default, which depends on whether the certificate is a certificate authorty or not, should be acceptable for most situations.
- `File path to save request` – this prompt appears only if you specify the -r option. Supply a location and file name for the PCKS10 certificate request. Supply a location and file name in which to save the certificate. The certificate is not saved unless you specify a location and file name.
- `Enter file path to save private key` – supply a location and file name in which to save the private key. Enter a password to protect private key. Optionally, supply a password with which to encrypt the private key. If you do not supply a password, the private key is not encrypted. This prompt appears only if you supplied a file in the previous prompt.
- `Enter file path to save identity` – supply a location and file name in which to save the identity. The identity file is a concatenation of the certificate, signer, and private key. This is the file that you supply to the server at start up. If the private key was not saved, createcert prompts for a password to save the private key. Otherwise, it uses the password provided earlier. The identity is not saved unless you provide a file name. If you do not save the identity file, you can manually concatenate the certificate, signer, and private key files into an identity file.

Examples

- **Example 1: Creating a signed certificate –** This example creates a signed certificate. No file name is provided for the signer's certificate, which makes it a self-signed root certificate.

```
<UnwiredPlatform_InstallDir>\UnwiredPlatform\Servers
\SQLAnywhere11\BIN32>createcert
SQL Anywhere X.509 Certificate Generator Version 11.0.1.2405
Enter RSA key length (512-16384): 1024
Generating key pair...
Country Code: US
State/Province: CA
Locality: Dublin
Organization: MyCompanyCA
Organizational Unit: PTO
Common Name: MyCompanyCA
Enter file path of signer's certificate:
Certificate will be a self-signed root
```

```
Serial number [generate GUID]:<enter>
Generated serial number: 3f52ee68c8604e48b8359e0c0128da5a
Certificate valid for how many years (1-100): 10
Certificate Authority (Y/N) [N]: Y
1.  Digital Signature
2.  Nonrepudiation
3.  Key Encipherment
4.  Data Encipherment
5.  Key Agreement
6.  Certificate Signing
7.  CRL Signing
8.  Encipher Only
9.  Decipher Only
Key Usage [6,7]: <enter>
Enter file path to save certificate: rsa_root.crt
Enter file path to save private key: rsa_key.key
Enter password to protect private key: <MyPwd>
Enter file path to save identity: id.pem
```

- **Example 2: Generating an enterprise root certificate –** To generate an enterprise root certificate (a certificate that signs other certificates), a self-signed root certificate should be created with a CA. The procedure is similar to Example 1. However, the response to the CA prompt should be `yes` and choice for roles should be option `6,7` (the default).

```
Certificate Authority (Y/N) [N]: y
1.  Digital Signature
2.  Nonrepudiation
3.  Key Encipherment
4.  Data Encipherment
5.  Key Agreement
6.  Certificate Signing
7.  CRL Signing
8.  Encipher Only
9.  Decipher Only
Key Usage [6,7]: 6,7
```

See also

- *Creating a Certificate Authority* on page 90
- *Generating a Certificate Request* on page 91
- *Generating an Afaria Certificate* on page 100
- *Sharing an Unwired Server Certificate with Afaria Server* on page 101

Key Creation (createkey) Utility

Creates RSA and ECC key pairs for use with Unwired Server end-to-end encryption. This utility is located in `<UnwiredPlatform_InstallDir>\Servers\SQLAnywhere11\BIN32`.

Syntax

```
createkey
```

When you run **createkey**, you are prompted for the following information:

- `Choose encryption type` – Choose RSA.
- `Enter RSA key length (512-16384)` – this prompt appears only if you chose RSA encryption. Choose a length between 512 bits and 16384 bits.
- `Enter file path to save public key` – specify a file name and location for the generated PEM-encoded public key. This file is specified on the MobiLink client by the e2ee_public_key protocol option.
- `Enter file path to save private key` – specify a file name and location for the generated PEM-encoded private key. This file is specified on the MobiLink server via the e2ee_private_key protocol option.
- `Enter password to protect private key` – optionally, supply a password with which to encrypt the private key. The private key is not encrypted if you do not supply a password. This password is specified on the MobiLink server via the e2ee_private_key_password protocol option.

Examples

- **Basic Example –** This example creates an RSA key pair:

```
>createkey
SQL Anywhere Key Pair Generator Version 11.0.0.2376
Enter RSA key length (512-16384): 2048
Generating key pair...
Enter file path to save public key: rsa_key_public.key
Enter file path to save private key: rsa_key_private.key
Enter password to protect private key: pwd
```

See also

- *Configuring Unwired Server Administration Certificates* on page 85

Key Tool (keytool) Utility

The key tool is a Java development kit utility that allows you to manage a keystore (database) of private keys and their associated X.509 certificates. The keytool utility also manages certificates from trusted entities. This utility is located in `<UnwiredPlatform_InstallDir>\UnwiredPlatform\Servers\SQLANywhere11\Sun\JRE160_x86\bin`.

keytool enables users to create and manage their own public/private key pairs and associated certificates for use in self-authentication or data integrity and authentication services, using digital signatures. It also allows users to cache the public keys (in the form of certificates) of their communicating peers.

Syntax

```
keytool -list | -printcert | -import | -export| -delete | -selfcert |
-certreq | -genkey [options]
```

Parameters

- **-list –** displays the contents of a keystore or keystore entry.
- **-printcert –** displays the contents of a certificate stored in a file. Check this information before importing a certificate as a trusted certificate. Make sure certificate fingerprints as as expected.
- **-import –** imports a certificate to:

  1. Add a certificate or certificate chain to the list of trusted certificates, or
  2. Import a certificate reply received from a certificate authority (CA) as the result of submitting a certificate signing request (CSR).

  The value of the `-alias` option indicates the type of import you are performing. If the alias exists in the database, then it is assumed you want to import a certificate reply. **keytool** checks whether the public key in the certificate reply matches the public key stored with the alias, and exits if they do not match. If the alias identifies the other type of keystore entry, the certificate is not imported. If the alias does not exist, it is created, and associated with the imported certificate.
- **-export –** exports a certificate to a file.
- **-delete –** deletes a certificate from the list of trusted certificates.
- **-selfcert –** generates a self-signed certificate. The generated certificate is stored as a single-element certificate chain in the keystore entry identified by the specified alias, where it replaces the existing certificate chain.
- **-certreq –** generates a Certificate Signing Request (CSR), using the PKCS #10 format. A CSR is intended to be sent to a CA, which authenticates the certificate requestor and returns a certificate or certificate chain, used to replace the existing certificate chain in the keystore.
- **-genkey –** generates a key pair (a public key and associated private key). Wraps the public key into an X.509 v1 self-signed certificate, which is stored as a single-element certificate chain. This certificate chain and the private key are stored in a new keystore entry identified by `<alias>`.
- **[*options*] – these options are available with the -genkey parameter:**

Option	Description
`-keystore <keyStoreLocation>`	The name and location of the persistent keystore file for the keystore managed by **keytool**. If you specify, in the `-keystore` option, a keystore that does not exist, that keystore is created. If you do not specify a `-keystore` option, the default keystore is a file named `.keystore` in your home directory. If that file does not exist, it is created.

Option	Description
`-storepass <password>`	The password that is used to protect the integrity of the keystore. The password must be at least 6 characters long. It must be provided to all commands that access the keystore contents. For such commands, if a `-storepass` option is not provided at the command line, the user is prompted for it.
`-file <certificateFile>`	The certificate file location.
`-noprompt`	During import, removes interaction with the user.
`-trustcacerts`	When importing a certificate reply, the certificate reply is validated using trusted certificates from the keystore, and using the certificates configured in the `cacerts` keystore file. This file resides in the JDK security properties directory, `java.home \lib\security`, where java.home is the runtime environment's directory. The `cacerts` file represents a system-wide keystore with CA certificates. System administrators can configure and manage that file using **keytool**, specifying "jks" as the keystore type.
`-alias <alias>`	The logical name for the certificate you are using.
`-keypass <password>`	The password used to protect the private key of the key pair. If you press enter at the prompt, the key password is set to the same password as for the keystore. `keypass` must be at least 6 characters long.

Examples

- **Example 1: Viewing the keystore –** Display the contents of the keystore:

  ```
  keytool -list -keystore <filePath>\keystore.jks -storepass
  <storepass>
  ```

- **Example 2: Importing a certificate reply from a CA –** Import a certificate:

  ```
  keytool -import -file <certificate file> -keystore
  <filePath>\keystore.jks -storepass <storepass> -noprompt -
  trustcacerts -alias <alias>
  ```

- **Example 3: Deleting a certificate –** Delete a certificate:

  ```
  keytool -delete -alias <alias> -keystore <filePath>
  \keystore.jks -storepass <storepass>
  ```

- **Example 4: Generating a key pair –** Create a new key entry in a keystore:

  ```
  keytool -genkey -keystore <filePath>\keystore.jks
  ```

The certificate request must be signed by a CA. Alternatively, you can self-sign the certificate by using the -selfcert **keytool** option.

Unwired Server Runtime Utilities

Unwired Server runtime supports many utilities used to manage the server environment and its artifacts.

Launch these utilities from the command line, or from Sybase Control Center.

Unwired Server Service (sup-server-service) Utility

Installs, removes, starts, and stops Unwired Server as a Windows service from the command line. This utility is located in `<UnwiredPlatform_InstallDir>\Servers\UnwiredServer\bin`.

Prerequisites: You need JDK 1.6.0_16 or higher to use this utility. If you choose to use an existing JDK option during installation, also ensure that you have set the `JAVA_HOME` environment variable to the installation location of this JDK version. Otherwise, you must use a text editor to modify the existing JDK path in the batch file itself.

Syntax

```
sup-server-service [install auto | install manual | remove | start |
stop]
```

Parameters

- **install auto –** installs Unwired Server as a Windows service in auto-start mode.
- **install manual –** installs Unwired Server as a Windows service in manual start mode.
- **remove –** uninstalls the Unwired Server service.
- **start –** starts the Unwired Server service.
- **stop –** stops the Unwired Server service.

Examples

- **–** This command installs Unwired Server as an auto-start Windows service:

  ```
  sup-server-service   install auto
  ```

Usage

Use the command line utility only when you cannot start or stop the service using the Unwired Server desktop shortcuts.

See also

- *Preparing Unwired Server to Connect to JDBC Databases* on page 31

Runtime Configuration (configure-mms) Utility

Applies manual Unwired Server configurations. Use the `configure-mms.bat` utility before restarting Unwired Server. This utility is located in `<UnwiredPlatform_InstallDir>\Servers\UnwiredServer\bin`.

Syntax

```
configure-mms [<clusterName>      ]
```

Parameters

- **clusterName –** the name of the cluster to which the Unwired Server belongs.

Usage

In situations that require server configuration changes to be performed outside of Sybase Control Center, use the `configure-mms.bat` utility to commit these changes.

Runtime Reconfiguration (reconfigure-mms) Utility

Reconfigures local or remote Unwired Servers to use different consolidated database (CDB) properties. You can enter new values for the CDB host, port, server name, user name, password, or start mode. This utility is located in `<UnwiredPlatform_InstallDir>\Servers\UnwiredServer\bin`.

This script updates connection values for the cluster database, endpoints and DSNs for the Unwired Server. If the CDB is on the same host (local), the script also recreates the CDB Windows service.

Before using this command:

1. If the CDB installed on a different host from Unwired Server, ensure that the CDB server is running.
2. If the CDB server is co-hosted with Unwired Server, ensure that all the services, including the CDB services, are stopped.

Syntax

```
reconfigure-mms.bat [-installtype {default | remote}] [-servicemode
{auto | manual}] [-cdbuser {cdb_user_name}] [-cdbpwd
{cdb_user_password}] [-cdbhost {cdb_host}] [-cdbport {cdbport}] [-
cdbserver {cdb_server_name}]
```

Parameters

- **installtype –** the type of database install; either `default` or `remote`. Use `remote` when moving database to different host than the one shared with Unwired Server

- **cdbhost –** the name of the machine where the existing database server is running.
- **cdbport –** the port over which CDB communication takes place. Default: 5200.
- **cdbpwd –** the database user password. Default: sql.
- **cdbserver –** the name of the database server used to manage requests for the consolidated database. Default: <MyComputer>_primary. If the database is on another host or is part of a cluster, you may need to use another host name.
- **cdbuser –** the user name to log into the CDB.
- **servicemode –** start mode of Unwired Server. Use `auto` to start Unwired Server in service mode. Use `manual` to start it in application mode.

Examples

- **Basic Example –** This command start the CDB at a new port of 5300:

```
reconfigure-mms.bat -cdbport 5300
```

- **Advanced Example –** This command changes the host, port, and server name of CDB because it has moved to a different host:

```
reconfigure-mms.bat -cdbhost W2K3-VM -cdbport 5200 -cdbuser dba -
cdbpwd sql -cdbserver
W2K3-VM_primary
```

Usage

The usage varies depending on whether or not the CDB is co-hosted with Unwired Server:

- **When CDB is hosted separately –** Perform these steps:
  1. If you are moving the CDB to a new host, copy the `.db` and `.log` files from `<UnwiredPlatform_InstallDir>\UnwiredPlatform-XX\Servers\UnwiredServer\data` from the existing host to the new host.
  2. Start the CDB server.
  3. Run **reconfigure-mms.bat** by passing necessary property values as described in syntax.
  4. Restart Unwired Platform services.
- **When CDB is co-hosted –** Perform these steps:
  1. Stop all Unwired Platform services, including CDB services.
  2. Run **reconfigure-mms.bat** by passing necessary property values as described in syntax. At minimum, the *cdbhost* property is required.
  3. Restart Unwired Platform services.

License Upgrade (license) Utility

Upgrades the license in a served model when you purchase more licenses from Sybase. In an unserved model, you simply replace the license file. This utility is located in `<UnwiredPlatform_InstallDir>\Servers\UnwiredServer\bin`.

The `license.bat` utility upgrades only Unwired Platform licenses. You cannot use it to upgrade Afaria licenses, which can be upgraded only with the Afaria Administrator.

Syntax

```
license.bat <PE> <LT> [licenseNumber]
```

Parameters

- **PE –** use the corresponding abbreviation that matches the product edition in your license. Valid product editions include:
  - EE for Enterprise Edition
  - AE for Advanced Deployment Edition
  - SK for Starter Deployment Edition
  - ED for Enterprise Developer Edition
  - PD for Personal Developer Edition
- **LT –** use the corresponding abbreviation that matches the license type in your license. Valid license types include:
  - AC for Application deployment CPU license
  - AS for Application deployment seat license
  - CP for CPU license
  - ST for Seat license
  - OT for Other license
  - SS for Standalone seat license
  - DT for Development and test license

Examples

- **Basic Example –** If your license includes an uncommented line that reads `INCREMENT SUP_BASESRVR SYBASE 2010.03316 31-mar-2010 uncounted \ VENDOR_STRING=PE=EE;LT=CP HOSTID=ANY ISSUER="CO=Sybase, \`, use this command.

  ```
  license.bat EE CP <MyLicenseNumber>
  ```

Usage

Depending on the product edition you have, the license type varies according to these guidelines:

- If you entered EE for product edition, the license type can be AC, AS, CP, or ST.
- If you entered PD for product edition, the license type can be SS.
- If you entered ED for product edition, the license type can be DT.

See also

- *Updating and Upgrading Unwired Platform Licenses* on page 278
- *Locating Information in a License File* on page 279

Synchronization Monitor (mlmon) Utility

Monitors the progress of replication-based synchronization (RBS) and checks for potential data contention or bottlenecks. This utility is located in `<UnwiredPlatform_InstallDir>\Servers\SQLAnywhere11\BIN32`.

Prerequisites: You need JDK 1.6.0_16 or higher to use this utility. If you choose to use an existing JDK option during installation, also ensure that you have set the `JAVA_HOME` environment variable to the installation location of this JDK version. Otherwise, you must use a text editor to modify the existing JDK path in the batch file itself.

Syntax

```
mlmon [connect-options|inputfile.{mlm|csv}]
```

When you execute this command, you are prompted to provide:

- Valid administrator credentials
- RBS protocol (default: HTTP)
- RBS port (default: 2480)

Parameters

- **connect-options –** allows you to connect to a messaging server on startup. A monitor connection starts like a synchronization connection to the messaging server. Allowed values are:

Option	Description
`-u ml_username`	Required to connect to the messaging server.
`-p password`	Required to connect to the messaging server.

Option	Description

`-x [tcpip | tls | http | https] [(keyword=value;...)]`	Required to connect to the messaging server. The keyword=value pairs can be:
	• Host – the network name or IP address of the computer where the messaging server is running. By default, it is the computer where the monitor is running.
	• Protocol – should be set to the same network protocol and port as the messaging server is using for synchronization requests.
	• Additional Network Parameters – optional parameters, including:
	  • `buffer_size=number`
	  • `client_port=nnnn`
	  • `client_port=nnnn-mmmmm`
	  • `persistent=[0|1]` (HTTP and HTTPS only)
	  • `proxy_host=proxy_hostname` (HTTP and HTTPS only)
	  • `proxy_port=proxy_portnumber` (HTTP and HTTPS only)
	  • `url_suffix=suffix` (HTTP and HTTPS only)
	  • `version=HTTP-version-number` (HTTP and HTTPS only)

`-o outputfile.{mlm|csv}`	Closes the monitor at the end of the connection and saves the session in the specified file.

- **inputfile.{mlm|csv} –** prompts the monitor to open the specified file.

Package Administration Utilities

Deploy, delete, import, and export application packages using the administration command line utilities. You can execute these commands from the administration command line utility console or from the command line.

To issue commands using the interactive administration command line utility console, open `<<UnwiredPlatform_InstallDir>>\Servers\UnwiredServer\bin\supadmin.bat`. You must enter your host name, port, and administrator credentials before entering a command. Otherwise, the console prompts you for:

- Host – the host name for the Unwired Server. The default value is localhost.
- Port – the port used for the Unwired Server. The default value is 2000.
- Login ID – the administrator login ID to perform authentication with. The default value is supAdmin.

- Password – the administrator password to perform authentication with. The default value is s3pAdmin.

To issue the utility at the command line use:

```
supadmin.bat -host host name -port port -u username -pw password
commandName commandOptions
```

For example, to delete the package `test:1.0` from the domain, enter:

```
supadmin.bat -host test01.sybase.com -port 2000 -u supAdmin -pw
s3pAdmin delete -d default -pkg test:1.0
```

Deploy Application Package (deploy) Utility

Deploys a package to Unwired Server with the administration client APIs.

Syntax

```
deploy [deploy-options]
```

Parameters

- **deploy-options –** uses these option to deploy and configure the package:

Option	Description
-d *domain*	Specifies the domain to which the package belongs.
-dcf *descriptorFile*	(Optional) Specifies the descriptor file for the package.
-dm *deploymentMode*	Sets the deployment mode. The mode determines how deployment handles the objects in a deployment unit and package. Allowed values are: • `UPDATE` – updates the target package with updated objects. After deployment, objects in the server's package with the same name as those being deployed are updated. • `NOCLOBBER` – deploys the package only if there are no objects in the target server's package that have the same name as any of those objects being deployed. • `REPLACE` – replaces any of the target objects with those in the package. After deployment, the servers package contains only those objects being deployed. • `VERIFY` – do not deploy package. Only return errors, if any. Used to determine the results of the `UPDATE` deploy mode.
-file *deploymentUnitFileName*	Defines the file name of the deployment unit. For example, `MyDeployUnit.xml`.

Option	Description
-rm *roleMapping*	Defines the role mapping for the package. Accepted values are: • `role1=AUTO` – the logical role defined in the package. • `role2=SUPAdmin,SUPUser` – the physical roles defined in the security provider. The role mapping `-rm role1,role2` maps the logical roles of the package to the physical roles in the server.
-sc *securityConfig*	Defines the security configuration for the package. Select an existing security configuration from the domain to which the package will be deployed.
-sl	Enables silent mode, which disables all user interactive questions during package deployment. During silent mode, the default values for each option are used. This mode is mainly used when writing a batch executing file. For example, the command line `deploy -file deployunit.xml -sl` deploys the package to the default domain with a deploy mode of `UPDATE` and a sync mode of `REPLICATION`, without asking for user confirmation.
-sm *syncMode*	Defines the synchronization mode for the package; either `MBS` for messaging-based synchronization, or `REPLICATION` for replication-based synchronization.

Examples

- **Basic Example –** This command updates an existing deployment unit called `samples/uep/deployment/customer_list_unit.xml`. The batch file uses default values for all other command line options:

```
deploy   -file samples/uep/deployment/customer_list_unit.xml -dm
UPDATE -sl
```

Import Application Package (import) and Export Application Package (export) Utilities

The **import** command imports an existing package to Unwired Server. The **export** command exports a package from Unwired Server.

Syntax

```
import [import-options]
```

```
export [export-options]
```

Parameters

- **import-options | export-options –** use these options to import or export the package:

Option	Usage	Description
-d *domain*	import, export	During import, specifies the domain to which you are importing the package. During export, specifies the domain to which the package currently belongs.
-file *fileName*	import	Specifies the file name of the exported package.
-pkg *packageName*	export	Specifies the name of the package to export.

Examples

- **Import example –** This command imports the "samples/uep/deployment/customer_list_unit_copy.xml" file to the domain "default":

```
import   -d default -file samples/uep/deployment/
customer_list_unit_copy.xml
```

- **Export example –** This command exports the package "samplePkg" from the domain "default" to make it available to other domains:

```
export   -d default -pkg samplePkg
```

Delete Application Package (delete) Utility

Deletes a package from Unwired Server.

Syntax

```
delete [delete-options]
```

Parameters

- **delete-options –** use these options to delete the package:

Option	Description
-d *domain*	Specifies the domain to which the package belongs.
-pkg *packageName*	Specifies the name of the package to delete.

Examples

- **–** This command deletes samplePkg from Unwired Server:

```
delete   -d default -pkg samplePkg
```

Start and Stop sampledb Server (sampledb) Utility

Starts and stops the sampledb server from the command line.

Syntax

```
sampledb.bat [install auto | install manual | start | stop ]
```

Parameters

- **install auto –** installs the sampledb server as a Windows service in auto-start mode.
- **install manual –** installs the sampledb server as a Windows service in manual start mode.
- **start –** starts the sampledb server.
- **stop –** stops the sampledb server.

Advantage Database Server Backup (adsbackup) Utility

A command line backup utility for use with Advantage Database Server. The **adsbackup** utility collects all the required information to perform a backup or restore, and executes the backup or restore on the specified server. This utility is located in `<UnwiredPlatform_InstallDir>\Servers\Advantage910\Server`.

Prerequisites: The **adsbackup.exe** utility loads `ACE32.DLL` or `libace.so` dynamically so it can be used with different versions of Advantage Client Engine (ACE). The minimum ACE version requirement is v6.0. However, if the server operating system is NetWare, the ACE version must be v7.0 or greater.

Syntax

To back up a directory of free tables:

```
adsbackup.exe [options] <src path> <file mask> <dest path>
```

To back up a data dictionary and its associated tables:

```
adsbackup.exe [options] <src database> <dest path>
```

Parameters

- **src path –** the path to the free tables you want to back up.
- **file mask –** a file mask similar to "*.adt" or "*.dbf" which determines which tables to include in the backup image.
- **dest path –** the backup image destination.
- **src database –** the path to the data dictionary you want to back up.
- **[options] –** these options are available:

Option	Description
-a	Creates a new backup image and initializes the source data so it can be used in a subsequent differential backup.
-c [ANSI|OMI]	Sets the character type of the SQL statement or connection used to open each table and transfer its data. Default: ANSI.
-d	Logs a warning before overwriting existing tables. The default behavior is to overwrite all tables when performing a backup or restore operation.
-f	Performs a differential backup. Includes only tables and records that have been modified since the last backup. Use this option only on databases on which you have previously called **ads-backup -a** to initialize the differential backup.
-h [ON|OFF]	Sets the rights-checking mode (used to open each table and transfer its data) of the SQL statement or connection used by the backup utility.
-i *<file1, file2...>*	Provides a list of tables to include in the backup or restore command. When using a data dictionary, specify the table name in the dictionary. When using free tables, specify the base table name including the file extension; for example, "table1.adt".
-e *<file1, file2...>*	Provides a list of tables to exclude from the backup or restore command. When using a data dictionary, specify the table name in the dictionary. When using free tables, specify the base table name including the file extension; for example, "table1.adt".
-m	Makes a backup or restore copy of only the data dictionary. Therefore, only metadata is backed up.
-n *<path>*	Specifies the location in which to store the backup log table.
-o *<filepath>*	Specifies a file name, along with the file path to the backup log table location.
-p	If the source path is a database, indicates the database password for the user. If the source path is a directory, lists the free table passwords. Free table usage can pass a single password for all encrypted tables, or a name=value pair for each table; for example, "password_for_all" or "table1=pass1;table2=pass2".
-q [PROP|COMPAT]	Specifies the locking mode of the SQL statement or connection used by the backup utility. This is the locking mode used to open each table and transfer its data; either proprietary (PROP) or compatible (COMPAT).
-r	Performs a restore instead of a backup (which is the default behavior).

Option	Description
-s *<server path>*	Specifies the connection path of the ADS server to perform the backup or restore operation. In most situations, **adsbackup** can determine the connection path to use by examining the source path. Use this option if you are having connection problems or want to use a specific connection path when performing the backup or restore.
-t *<server port>*	Defines the server port **adsbackup** connects to when using the Advantage JDBC Driver. This option is valid only with the Java **adsbackup** utility. Default: 6262.
-u [ADT|CDX|NTX]	Sets the table type of the SQL statement or connection used by the backup utility. This is the table type used to open each table and transfer its data. This also determines the table type of the log file produced by the backup or restore procedure.
-v [1-10]	Configures the lowest level of error severity to return. Default: 1.

Usage

You can use **adsbackup** in conjunction with the Unwired Server database file recovery tool **MORecover**. See *Sybase Unwired Platform Administration Guide > System Reference > Command Line Utilities > Unwired Server Runtime Utilities > Unwired Server Database File Recovery (MORecover) Utility*.

For more information on **adsbackup.exe** usage and sample use cases, see the *Advantage Database Server* documentation.

See also

- *Backing Up Messaging Data* on page 271

Unwired Server Database File Recovery (MORecover) Utility

The **MORecover** utility, with **adsbackup.exe**, backs up Unwired Server database files while the server is running. This utility is located in `<UnwiredPlatform_InstallDir>\Servers\MessagingServer\Bin`.

Recent versions of the Advantage Database Server do not allow Unwired Server database files to be copied while the service is running. Stopping Unwired Server to perform regular backups is highly inconvenient in a production environment.

Prerequisite: Before executing **MORecover**:

1. Create an MORecover snapshot by running **adsbackup** to back up the tables required by MORecover. Use the **-i** (include) option to indicate the database tables to include in the backup. For an MORecover snapshot, these tables are:

- APPLICATIONS
- CFG_IDS
- CFG_PROP_VALUES
- CFG_SUBFOLDER_PROP_VALUES
- CFG_TEMPLATES
- DEVICES
- USER_DEVICE
- USERS

The source database argument for the command must be the full path of the `OBR.add` file in the Unwired Server data folder. The destination can be any folder. A collection of files constituting the backed-up data is created in the location you specify and stored as the MORecover backup data (see Example 1 below).

2. Convert the files generated by **adsbackup** into a standard database format. The generated files reflect the raw data from the original tables, excluding the index data. Therefore, files generated by **adsbackup** cannot be used directly as a database (see Example 2 below).

For information on **adsbackup** utility parameters and options, see *System Administration guide > System Reference > Command Line Utilities > Unwired Server Runtime Utilities > Advanatage Database Server Backup (adsbackup) Utility*.

Syntax

```
MORecover <recoveryDatabaseLocation>
```

Examples

- **Example 1: –** This example uses **adsbackup -i** (include) to indicate the database tables to include in the MORecover snapshot.

  ```
  adsbackup.exe -i
  APPLICATIONS,CFG_IDS,CFG_PROP_VALUES,CFG_SUBFOLDER_PROP_VALUES,CF
  G_TEMPLATES,DEVICES,USER_DEVICE,USERS <sourceDatabaseLocation>
  <destinationFileLocation>
  ```

- **Example 2: –** Before running MORecover to restore your database, you must convert the data files created by **adsbackup** back to a standard database format. The following example uses the **-r** option to restore the backup data to a temporary folder. You can then point MORecover to the restored data in this folder.

  ```
  adsbackup.exe -r <destinationFileLocation>
  <recoveryDatabaseLocation>
  ```

Usage

While the specified subset of tables typically results in a time-efficient backup operation, Sybase recommends that you schedule this command to run regularly during a time of light system load, since the backup operation contends with Unwired Server normal processing for database resources.

See also

- *Restoration of the Messaging Data* on page 273

Update Properties (updateprops.bat) Utility

Performs multiple functions, including registering or removing a participating node for a cluster, or update a specific server property.

Syntax

```
updateprops.bat [-u username] [-p password] [-d dsn] [-f
propertyFile]
[-cn clusterName] [-nv "<propertyName=NewValue>"] [-r] [-v] [-x]
```

Parameters

- **-u *username* –** the platform administrator username.
- **-p *password* –** the platform administrator password.
- **-d *dsn* –** the data source name (DSN) of the cluster database.
- **-cn *clusterName* –** the name that identifies the Unwired Platform Cluster
- **-nv "<*propertyName=NewValue*>" –** one or more platform property values that requires change. Multiple values can be defined; however, they must be separated by the pound symbol (#). For example:

  ```
  –nv
  “ml.threadcount=10#sup.admin.port=2005#sup.sync.port=2490"
  ```
- **-v –** use verbose output in the command window.

Examples

- **Changing a cdb threadcount property –** Update the ml.threadcount property of the production environment consolidated database to `20` by running:

  updateProps.bat -nv "ml.threadcount=20"

  This is only recommended for deployment editions of Unwired Platform.

Usage

Before running this utility, ensure that the data tier is available; otherwise platform data is not modified correctly.

See also

- *Changing the Consolidated Database Server Thread Count and Pool Size* on page 26

Configuration Files

Configuration files hold the initial settings for various Unwired Platform components or subcomponents.

All Unwired Platform components read their configuration files at start-up. Some, like the relay server or cluster configuration, periodically check the configuration files for changes. For Unwired Server, administrators can instruct the server to reread the configuration files and apply the changes to the current process. However, some Unwired Server changes require that you restart the server.

Configure Unwired Server and its embedded subcomponents using Sybase Control Center, which then writes values to the appropriate file. For relay server, you must manually configure component files.

Unwired Server Configuration Files

Use Unwired Server configuration files to manually modify server security, logging, and subcomponent configurations.

Sybase recommends that you edit these `.properties` and `.xml` files only if you cannot use Sybase Control Center to configure Unwired Server properties.

See also

- *Sybase Control Center Configuration Files* on page 362
- *Relay Server Configuration Files* on page 363

Global Unwired Server Properties (sup.properties) Configuration File Reference

`sup.properties` is a global properties file that allows you to configure many subcomponents used by Unwired Server. This file is located in `<UnwiredPlatform_InstallDir>\Servers\UnwiredServer\Repository\Instance\com\sybase\sup\server\SUPServer`.

Property	Default	Description
sup.admin.port	2000	The port used by the embedded application server.
sup.admin.protocol	IIOP	The protocol used for server administration (read-only).
sup.admin.httpports	8000	The HTTP ports used by the embedded application server.

Property	Default	Description
sup.admin.httpsports	8001, 8002	The HTTPS (secure) ports used by the embedded application server.
sup.admin.iiopsport	2001	The IIOPS port used by the embedded application server.
sup.sync.sslkeystore	`Repository/Security/keystore.jks`	The relative path to the Unwired Server keystore file.
sup.sync.sslkeystore_password	changeit	The password to unlock the keystore file
sup.sync.ssltruststore	`Repository/security/truststore.jks`	The relative path to the Unwired Server truststore file.
sup.sync.ssltruststore_password	changeit	The password to unlock the truststore file
sup.sync.port	2480	The synchronization port.
sup.sync.httpsport	2481	The secure synchronization port.
sup.sync.protocol	HTTP	The protocol used for synchronization. By default the protocol is HTTP; however, you can also use HTTPS.
sup.sync.certificate	n/a	The fully qualified path to the certificate file.
sup.sync.certificate_password	n/a	The password to unlock the certificate.
sup.cluster.name	n/a	The Unwired Platform cluster name.
sup.user.options	n/a	The Unwired Server user startup options.
sup.install.number	<number>	The Unwired Platform install number.
sup.java.install	n/a	The Unwired Platform Java install number.
sup.host	<computerName>	The host name of the SUP server.
sup.install.asservice	false	Indicates whether or not Unwired Server is installed in service mode.
sup.node.status	resumed	The Unwired Platform cluster status: suspend, resume, pending, suspended, resumed, pending.
sup.node.home	n/a	Mobile messaging service home path.

Property	Default	Description
sup.imo.upa	n/a	Encrypted user name and password of admin@system.
sup.msg.outbound_queue_prefix	sup.mbs.moca.	The outbound queue prefix.
sup.msg.inbound_queue_prefix	sup.mbs.	The inbound queue prefix.
sup.msg.inbound_count	25	The number of inbound queues.
sup.msg.outbound_count	5	The number of outbound queues.
ml.threadcount	5	The synchronization threadcount. This value must be 5 units lower than sqlany.threadcount.
ml.servername	none	The Unwired Server name.
ml.cachesize	50M	The maximum size for the RBS synchronization server memory cache.
relayserver.webserver.token	none	The relay server-backed Web server token. If left empty, a random value is generated.
relayserver.token	none	Replication-based synchronization backend server token. If empty, a random value is generated.
relayserver.https_port	443	the secure HTTP port for relay server connections.
relayserver.http_port	80	the HTTP port for relay server connections.
relayserver.type	IIS	The relay server-hosted operating system type: IIS or Unix.
relayserver.farm_name	none	(Applies only until previous versions are upgraded.) the server farm for which the relay server manages requests. This property is case-sensitive. The configured value must match the value defined for the RSOE or the connection fails. This property identifies the RBS farm. If empty, the value *<ClusterName>*.SUPFarm is used.

Property	Default	Description
relayserver.msg.token	none	the security token used by the server node to authenticate the back-end server connection with relay server. Each node requires a unique token: specify a unique string to a maximum of 2048 characters.
relayserver.trusted_certs	none	The trusted certificate path relative to Unwired Server home.
relayserver.protocol	HTTP	The relay server protocol: HTTP or HTTPS.
relayserver.host	relayserver.sybase.com	either the relay server host name or the host name of the load balancer (if one is used in your environment). Device clients use the host name to establish connections. Other relay servers can use the host name to identify peers if a load balancer is not used.
relayserver.webserver.farm_name	none	(Applies only until previous versions are upgraded.) the server farm for which the relay server manages requests. This property is case-sensitive. The configured value must match the value defined for the RSOE or the connection fails. If empty, *<ClusterName>*.SUPWebServerFarm is used. This property identifies the Web server farm name used by DCN requests and DOE-C connections.
relayserver.msg.farm_name	none	(Applies only until previous versions are upgraded.) the server farm for which the relay server manages requests. This property is case-sensitive. The configured value must match the value defined for the RSOE or the connection fails. This property identifies the MBS farm. If empty, *<ClusterName>*.SUPMessagingFarm is used.
cdb.threadcount	20	The maximum number of tasks that the consolidated database server can execute concurrently.
cdb.databasename	default	The consolidate database name.
cdb.password	sql	The password for the consolidated database.
cdb.asa.mode	primary	The database mode when consolidate database type is SQL Anywhere (Sybase_ASA).
cdb.serverport	5200	The consolidated database port number.

Property	Default	Description
cdb.dsnname	default-cdb	The consolidated database DSN name.
cdb.install_type	default	The consolidated database install type.
cdb.username	dba	The consolidated database user name.
cdb.type	Sybase_ASA	The consolidated database type.
cdb.serverhost	none	The consolidated database server host name.
cdb.user.options	none	The consolidated database user-specified startup options.
cdb.servername	none	The consolidated database server name.
cldb.serverhost	none	The cluster database server host name.
cldb.serverport	5200	The cluster database port number.
cldb.username	dba	The cluster database user name.
cldb.password	sql	The cluster database password.
cldb.databasename	clusterdb	The cluster database name.
cldb.dsnname	none	The cluster database DSN name.
cldb.type	Sybase_ASA	The cluster database type.
monitoringdb.server-host	none	The monitoring database server host name.
monitoringdb.server-port	5200	The monitoring database port number.
monitoringdb.user-name	dba	The monitoring database user name.
monitoringdb.pass-word	sql	The monitoring database password.
monitoringdb.databa-sename	monitordb	The monitoring database name.
monitoringdb.type	Sybase_ASA	The monitoring database type.
cluster.version	1	The cluster version number.
cluster.sync.share-dpath	none	The shared data path for CSYNC/DSYNC zip files; required only when "cluster.sync.share-dpath.enabled" is set to true.
cluster.sync.share-dpath.enabled	False	Indicates whether or not to enable the cluster's optional "Shared Data Path" feature.

Property	Default	Description
license.product.edition	none	The Unwired Platform license edition.
license.type	none	The Unwired Platform license type.
client.licenses	none	The number of Unwired Platform device licenses.
client.url_suffix	none	A URL suffix that allows a device application client to connect to a relay server farm. The default value for Apache on Linux is `/srv/iarelayserver`, and the default value for IIS on Windows is `/ias_relay_server/server/rs_server.dll`. For example, to connect a client to a relay server farm called RBSFarm1, you might use `/sup_relay_server/client/rs_client.dll/RBSFarm1` as your URL suffix.
msg.http.server.ports	5001, 80	The Messaging Service HTTP ports.
msg.client.url_suffix	none	A URL suffix that allows a device application client to connect to a relay server farm. The default value for Apache on Linux is `/srv/iarelayserver`, and the default value for IIS on Windows is `/ias_relay_server/server/rs_server.dll`. For example, to connect a client to a relay server farm called MBSFarm1, you might use `/sup_relay_server/client/rs_client.dll/MBSFarm1` as your URL suffix.
msg.admin.webservices.port	5100	The Messaging Service administration port.
msg.rsoeOptions	none	The Messaging Service RSOE options. Used in the Messaging Service RSOE command line.
msgserver.location	none	The file location of the Messaging Service.

Property	Default	Description
webserver.rsoeOptions	none	(Applies only until previous versions are upgraded.) The RSOE options for the embedded Web server requests (for example, data change notifications and DOE-C).
webserver.client.url_suffix	none	A URL suffix that allows a device application client to connect to a relay server farm. The default value for Apache on Linux is `/srv/iarelayserver`, and the default value for IIS on Windows is `/ias_relay_server/server/rs_server.dll`.The client URL suffix for Web server requests when the relay server is configured (for example, data change notifications and DOE-C).
rsoeOptions	none	(Applies only until previous versions are upgraded.) The RSOE options for the Messaging Service. Used in the replication-based synchronization RSOE command line.
sqlany.mode	Primary	The consolidated database mode: only primary is supported.
mac.address	none	The Mac address of the system.

Runtime Message Tracing (TraceConfig.xml) Configuration File

Enables message tracing for various system components, and sets the tracing level. This file is located in `<UnwiredPlatform_InstallDir>\Servers\MessagingServer\Data`

The syntax is:

```
<TraceConfig>
            <Dir>\...UnwiredServer\logs</Dir>
            <Module Name="<moduleName>" Level="<traceLevel>"
Desc="<moduleDescription>" />
            ...
</TraceConfig>
```

The `<Dir>` property indicates the directory where tracing messages are stored when component tracing is enabled. The default location is `<UnwiredPlatform_InstallDir>\Servers\UnwiredServer\logs\<moduleName>`. Messages for each module are stored as .txt files in the corresponding folder.

The configurable properties are:

- Module name – the name of the system component.

- Level – the tracing level:

Level	Messages logged
All	Complete system information
Trace	Finer-grained informational events than debug
Debug	Very fine-grained system information, warnings, and all errors
Info	General system information, warnings, and all errors
Warn	Warnings and all errors
Error	Errors only
Off	None

- Desc – a unique identifying description for the module.

See also

- *Configuring Messaging and Mobile Workflow Runtime Logging* on page 255
- *Configuring Mobile Workflow Tracing* on page 256

Sybase Control Center Configuration Files

Use Sybase Control Center configuration files to configure Sybase Control Center services, plug-ins, logging, and security.

Sybase recommends that you edit these `.properties` and `.xml` files only if you cannot use Sybase Control Center to configure the properties.

See also

- *Unwired Server Configuration Files* on page 355
- *Relay Server Configuration Files* on page 363

Sybase Control Center Logging (log4j.properties) Configuration File

Enables and configures Sybase Control Center logging. The log4j configuration file is located in `<UnwiredPlatform_InstallDir>\SCC-XX\conf`.

log4j properties include:

Property	Default	Description
log4j.appender.agent.File	`<UnwiredPlatform_InstallDir>\SCC-XX\log\agent.log`	The name and location of the Sybase Control Center log file.

Property	Default	Description
log4j.appender.agent.MaxFileSize	5MB	The maximum size that a file can reach before a new one is created.
log4j.appender.agent.MaxBackupIndex	20	The number of log files that are backed up before the oldest file is deleted.

Role Mapping (roles-map.xml) Configuration File

Use the `<UnwiredPlatform_InstallDir>\SCC-XX\conf\roles-map.xml` to map roles for Sybase Control Center.

The `<uaf-roles>` section of the configuration file defines the available Sybase Control Center logical roles. The syntax is:

```
<uaf-roles>
   <role name="<myRoleName>" description="<myRoleDescription>"
</uaf-roles>
```

The `<security-modules>` section lists each login module defined in `csi.properties` file and maps the security provider's physical roles to the logical roles for Sybase Control Center. Specifically, the `SUP LDAP Login Module` default role mapping is:

```
<module name="SUP LDAP Login Module">
   <role-mapping modRole="SUP Administrator"
   uafRole="uaAnonymous,uaAgentAdmin,
   uaplug-inAdmin,sccAdminRole,sccUserRole" />
   <role-mapping modRole="SUP Domain Administrator"
   uafRole="uaAnonymous,uaAgentAdmin,uaplug-inAdmin,
   sccUserRole" />
 </module>
```

If necessary, replace the `modRole` values for each applicable login module with your own security provider role names to map them to Sybase Control Center roles. By default, the configuration file assumes that the security repository includes the "SUP Administrator" and "SUP Domain Administrator" roles. The "SUP Domain Administrator" role mapping is required only if you plan to assign domain administrators within the cluster.

Relay Server Configuration Files

Use relay server configuration files to set up the components of the relay server farm, and to define relay server host, port, protocol, and logging specifications.

You must manually configure the relay server using component files; you cannot access the server properties through the administration console.

See also

- *Chapter 5, Relay Server Clusters* on page 53

- *Relay Server Documentation* on page 54
- *Relay Server Utilities* on page 329
- *Configuring RSOE Logging* on page 258
- *Unwired Server Configuration Files* on page 355
- *Sybase Control Center Configuration Files* on page 362

Relay Server (rs.config) Configuration File

The rs.config file defines the relay server farm, including peer relay servers, back-end server farms and their server nodes.

Sybase recommends that you generate the file in Sybase Control Center. Then, if there are additional changes you need to make, you can implement those manually.

- For IIS: `C:\Inetpub\wwwroot\IAS_relay_server\Server\rs.config`
- For Apache: `<Apache_Home>/modules/rs.config`

This file contains these properties:

Definition	Property	Default	Possible values	Description
Relay server options				The setup options for the relay server.
	shared_mem	10	any integer	Specifies the maximum amount of shared memory that the relay server uses for state tracking. The default is 10 megabytes. This property is optional.
	start	No	auto, no, full path	Specifies how State Manager is started: • auto – The State Manager is started automatically using the State Manager command line defaults. • no — The State Manager is started externally as a Windows service. • full path — Specify the full path to the State Manager executable (rshost).

Definition	Property	Default	Possible values	Description
	verbosity	1	0 to log errors only. Use this logging level for deployment. 1 to set session-level logging. Provides an overview of a synchronization session. 2 to request-level logging. Provides a more detailed view of HTTP requests within a synchronization session.	Specifies the verbosity level for the relay server log.
Relay server peers				Identifies all hosts on which peer relay servers are installed. Each new definition starts with `[relay server]`. Add as many `[relay server]` sections as required for your deployment.
	enable	yes	yes|no	Specifies whether to allow connections to the peers.
	host	relay-server.sybase.com	Any string that does not use special characters or spaces.	either the relay server host name or the host name of the load balancer (if one is used in your environment). Device clients use the host name to establish connections. Other relay servers can use the host name to identify peers if a load balancer is not used.
	http_port	80	Any valid port number that is not already in use.	the HTTP port for relay server connections.
	https_port	443	Any valid port number that is not already in use.	the secure HTTP port for relay server connections.
	description	None	Any string.	description for the relay server.

Definition	Property	Default	Possible values	Description
Backend farm				IThe backend farm section specifies the properties of a back-end server farm. A back-end server farm is a group of homogenous back-end servers. A client making a request through the relay server farm must specify the back-end server farm it is targeting. There is one backend farm section for each back-end server farm. This section is identified by the `backend_farm` keyword.
	enable	yes	yes|no	Specifies whether to allow connections from this back-end server farm.
	id	None	Any string that does not use special characters or spaces.	the server farm for which the relay server manages requests. This property is case-sensitive. The configured value must match the value defined for the RSOE or the connection fails.
	client_security	on	on|off	(Optional). This property can only be set manually. For details on this property, see *http://infocenter.sybase.com/help/topic/com.sybase.help.sqlanywhere.11.0.1/mlserver_en11/ml-relayserver-config-file.html*.
	back-end_security	on	on|off	(Optional) This property can only be set manually. For details on this property, see *http://infocenter.sybase.com/help/topic/com.sybase.help.sqlanywhere.11.0.1/mlserver_en11/ml-relayserver-config-file.html*
	type	None	Any string	(Optional) This property can only be set manually. For details on this property, see *http://infocenter.sybase.com/help/topic/com.sybase.help.sqlanywhere.11.0.1/mlserver_en11/ml-relayserver-config-file.html*
	description	None	Any string	describes relay server farm usage.

Definition	Property	Default	Possible values	Description
Backend servers				Identifies backend Unwired Server synchronization components installed on the corporate LAN. Each new definition starts with `[backend_server]`. Add as many `[backend_server]` sections as required for your deployment.
	enable	yes	yes|no	Specifies whether to allow connections from this back-end server.
	farm	None	Any string that does not use special characters or spaces.	The ID of the farm the server node is part of.
	id	None	Any string that does not use special characters or spaces.	the node string that identifies the back-end replication or messaging based cluster. Combine one or more Unwired Servers to create a single server node. This property is case-sensitive. The configured value must match the value defined for the RSOE or the connection fails.
	mac	None	Any typical MAC address.	The MAC address of the server node. Only type a value if you want relay server to use MAC verification. Only define one MAC address. To determine what address to use, find the address in the `RSOE.log` file.
	token	None	Any string that does not use special characters or spaces.	the security token used by the server node to authenticate the back-end server connection with relay server. Each node requires a unique token: specify a unique string to a maximum of 2048 characters.
	description	None	Any string	(Optional) A description of the server node.

Comment out any of the optional properties with the pound sign (#) character.

To apply the changes you make to this file, run this command from the directory where the `rs.config` file is located:

```
rshost -f rs.config -q -qc -u
```

Example: N+2 Architecture Configuration

This file is an example of how a optimally redundant N+2 production relay server architecture is configured. There are two relay servers in the Relay Server farm: RS1.sybase.com and RS2.sybase.com. There are also two backend server types: Afaria and SUP. The Afaria system has two backend servers: Afaria1.sybase.com and Afaria2.sybase.com. The Unwired Platform system has one backend server: SUP.sybase.com.

Assume relay servers use Microsoft IIS, and that this Web server is configured with the following paths:

- `\ias_relay_server\client` – the install location of `rs_client.dll`.
- `\ias_relay_server\server` – the install location of `rs_server.dll`, `rshost.exe`, and `rs.config`.

`#----------------------------`
`# Relay server with auto start option`
`#----------------------------------------`
`[options]`
`start = no`
`verbosity = 1`
	Start State Manager externally as a Windows service.

`#--------------------`
`# Relay server peers`
`#--------------------`
`[relay_server]`
`enable = yes`
`host = RS1.sybase.com`
`http_port = 80`
`https_port = 443`
`description = Machine #1 in RS farm`
 	
`[relay_server]`
`enable = yes`
`host = RS2.sybase.com`
`http_port = 80`
`https_port = 443`
`description = Machine #2 in RS farm`
	Because there are two relay servers in the relay server farm, both instances need to be defined in the peer list. This list is used by the RSOE to establish a connection with each relay server node in the farm.

`#---------------`
`# Backend farms`
`#---------------`
`[backend_farm]`
`enable = yes`
`id = AfariaFarm`
`client_security = on`
`backend_security= on`
`description = This is the definition for the Afaria farm.`
 	
`[backend_farm]`
`enable = yes`
`id = SUPFarm`
`client_security = on`
`backend_security= on`
`description = This is the definition for the SUP farm.`
	There are two different types of farms in this example. One is an Unwired Platform farm, and the other is an Afaria farm. Each farm requires an entry in the backend farms section, and each must have a unique Farm ID.

`#-----------------`
`# Backend servers`
`#-----------------`
`[backend_server]`
`enable = yes`
`farm = AfariaFarm`
`id = for Afaria1.sybase.com`
`mac = 01-23-45-67-89-ab`
`token = 7b2493b0-d0d4-464f-b0de-24643e1e0feb`
 	
`[backend_server]`
`enable = yes`
`farm = AfariaFarm`
`id = for Afaria1.sybase.com`
`mac = 01-23-45-67-89-ac`
`token = de1aac83-a653-4e0f-8a6c-0a161a6ee407`
 	
`[backend_server]`
`enable = yes`
`farm = SUPFarm`
`id = for SUP.sybase.com`
`mac = 01-23-45-67-89-ad`
`token = 621ece03-9246-4da7-99e3-c07c7599031c`
	There are three backend server nodes in this scenario. Two of the backend servers are part of the AfariaFarm and the third backend server is part of the SUPFarm. Afaria backend servers use the Transmitter ID as the Server ID. Unwired Platform typically uses the machine name as the Server ID. The Server ID must be unique for each backend server entry.

Note: When the relay server architecture has been deployed and configured, clients would connect to these severs differently:

- For AfariaFarm connections use: url_suffix=/ias_relay_server/client/rs_client.dll/AfariaFarm
- For SUPFarm connections use: url_suffix=/ias_relay_server/client/rs_client.dll/SUPFarm

See also

- *Configuring a Multinode Relay Server Cluster with Sybase Control Center* on page 67
- *Configuring and Enabling Relay Server Logging* on page 258

Relay Server Outbound Enabler (rsoeconfig.xml) Configuration File

The RSOE configuration file holds the connection information to relay servers deployed on your network.

There are two ways to create this file:

- (Recommended) Use Sybase Control Center to generate the file from properties. Alternately you can export this configuration with regRelayServer script. The generated or exported version of the file has the correct IDs (that is, primary keys of the cluster database) set for each element. With either output, you can then modify the file as needed.
- Edit `<UnwiredPlatform_InstallDir>\Servers\UnwiredServer\config\rsoe.config.template.xml`, then apply the file with this command:

  ```
  <UnwiredPlatform_InstallDir>\UnwiredPlatform\Servers
  \UnwiredServer\bin
  \regRelayServer.bat<UnwiredPlatform_InstallDir>
  \UnwiredPlatform\Servers\UnwiredServer\config
  \rsoeconfig.xml
  ```

  This command also writes the configuration back to the cluster database.

Elements and their respective properties are alphabetically listed.

Note: If a parent element has an ID of 0 or has no attribute, all its descendants' IDs will be treated as 0 as well due to XML inheritance rules. Set ID attributes carefully to avoid unexpected results.

Element	Property	Description
<relayServers>		The parent element for all defined <relayServer> child elements.
	xmlns	The XML namespace used for this configuration file.
<relayServer>		A relay server definition.

Element	Property	Description
	host	either the relay server host name or the host name of the load balancer (if one is used in your environment). Device clients use the host name to establish connections. Other relay servers can use the host name to identify peers if a load balancer is not used.
	ID	The primary key used to identify the relay server in the cluster database. If ID is a positive integer, you are configuring an existing entry. If the ID is 0 or empty, you are configuring a new entry.
	port	the HTTP port for relay server connections.
	securePort	the secure HTTP port for relay server connections.
	urlSuffix	A URL suffix that allows a device application client to connect to a relay server farm. The default value for Apache on Linux is `/srv/iarelayserver`, and the default value for IIS on Windows is `/ias_relay_server/server/rs_server.dll`.
<unwiredServerFarm>		An Unwired Server farm definition. This definition includes <description> and <serverNode> child elements.
	ID	The primary key used to identify the Unwired Server farm in the cluster database. If ID is a positive integer, you are configuring an existing entry. If the ID is 0 or empty, you are configuring a new entry.
	name	the server farm for which the relay server manages requests. This property is case-sensitive. The configured value must match the value defined for the RSOE or the connection fails.
	type	the type of request managed by the relay server: replication based synchronization (RBS) or messaging based synchronization (MBS).
<description>		describes relay server farm usage.

Element	Property	Description
<serverNode>		The server node definition. This definition should at least one <rsoe> child element.
	ID	The primary key used to identify the server node in the cluster database. If ID is a positive integer, you are configuring an existing entry. If the ID is 0 or empty, you are configuring a new entry.
	name	the node string that identifies the backend replication or messaging based cluster. Combine one or more Unwired Servers to create a single server node. This property is case-sensitive. The configured value must match the value defined for the RSOE or the connection fails.
	token	the security token used by the server node to authenticate the back-end server connection with relay server. Each node requires a unique token: specify a unique string to a maximum of 2048 characters.

Monitoring Database Schema

The monitoring database includes several tables from which information is retrieved by Sybase Control Center.

These system tables are accessible to any user. The contents of monitoring tables can be changed only by Unwired Platform components.

You can browse the contents of these tables by using any database browsing tool, for example, Sybase Central.

See also

- *Status and Performance Monitoring* on page 210
- *Configuring Monitoring Performance Properties* on page 214
- *Deploying the Monitoring Database* on page 28

mms_rbs_request Table

Detailed history information for replication-based synchronization.

Column name	Column type	Description
id	integer	The identifier of the data row.
summaryId	varchar(50)	The identifier for the row summary information.
deviceId	varchar(255)	The identifier of the physical device performing the synchronization request.
userName	varchar(255)	The name of the user associated with the device ID.
packageName	varchar(255)	The package name of the MBO data synchronization or operation replay activity.
domain	varchar(255)	The domain to which the package involved in synchronization or operation replay belongs.
startTime	timestamp	The date and time the synchronization request was initiated.
endTime	timestamp	The date and time the synchronization request was completed.
syncTime	integer	The total time of the synchronization or operation replay activity, in milliseconds.
sendRows	integer	The number of rows downloaded during the mobile business object (MBO) synchronization. If 1 appears, the action was an operation replay.
isError	bit	Whether an error has occured during the synchronization or replay: 1 if errors were recorded, 0 if no errors were recorded.
sentBytes	integer	The number of bytes downloaded in the transaction.
receivedBytes	integer	The number of bytes uploaded in the transaction.

Column name	Column type	Description
syncPhase	varchar(20)	The current synchronization activity: upload or download. During upload, a client initiates operation replays to execute MBO operations on the back-end system. During download, a client synchronizes with Unwired Server to receive the latest changes to an MBO from the back-end system.
mboNames	varchar(500)	The MBO that downloaded information.
operationNames	varchar(500)	The operation replay.
operationReplays	integer	The number of operation replays performed. Zero (0) indicates that information was downloaded by the MBO during a synchronization action.
isMonitored	bit	Whether the MBO is monitored. If 1, it is monitored. If 0, it is not.
isLogged	bit	Whether the domain is logging data. If 1, domain is logging data. If 0, it is not.

mms_rbs_request_summary Table

Summary history information for replication-based synchronization transactions.

Column name	Column type	Description
id	integer	The identifier of the data row.
deviceId	varchar(255)	The identifier of the physical device performing the synchronization request.
userName	varchar(255)	The name of the user associated with the device ID.
packageName	varchar(255)	The package name of the mobile business object (MBO) data synchronization or operation replay activity.
domain	varchar(255)	The domain to which the package involved in synchronization or operation replay belongs.
startTime	timestamp	The date and time the synchronization request was initiated.

Column name	Column type	Description
endTime	timestamp	The date and time the synchronization request was completed.
request syncTime	integer	The total time of the synchronization or operation replay activity, in milliseconds.
totalReceivedRows	integer	Always 1.
totalErrors	integer	The number of all exceptions during the synchronization request.
totalsentBytes	integer	The number of all bytes dowloaded by the MBO.
totalreceivedBytes	integer	The number of all bytes uploaded by the MBO.
totalOperationReplays	integer	The number of all operation replays for the MBO.
isMonitored	bit	Whether the MBO is monitored. If 1, it is monitored. If 0, it is not.
isLogged	bit	Whether the domain is logging data. If 1, domain is logging data. If 0, it is not.

mms_rbs_mbo_sync_info Table

A subset of the mms_rbs_request table data.

Column name	Column type	Description
id	integer	The identifier of the data row.
packageName	varchar(255)	The package name of the mobile business object (MBO) data synchronization.
domain	varchar(255)	The domain to which the package involved in synchronization belongs.
mboName	varchar(255)	The name of the MBO performing the transaction.
startTime	timestamp	The date and time the synchronization request was initiated.
endTime	timestamp	The date and time the synchronization request was completed.

Column name	Column type	Description
syncTime	integer	The total time of the synchronization, in milliseconds.
isError	bit	Whether errors have occured during the synchronization: 1 if errors were recorded, 0 if no errors were recorded.

mms_rbs_operation_replay Table

A subset of the mms_rbs_request table data.

Column name	Column type	Description
id	integer	The identifier of the data row.
deviceId	varchar(255)	The identifier of the physical device performing the operation replay.
userName	varchar(255)	The name of the user associated with the device ID.
packageName	varchar(255)	The package name of the mobile business object (MBO).
domain	varchar(255)	The domain to which the package involved in synchronization or operation replay belongs.
startTime	timestamp	The date and time the operation replay request was initiated.
endTime	timestamp	The date and time the operation replay was completed.
processTime	integer	The total time of the operation replay, in milliseconds.
mbo	varchar(255)	The MBO performing the transaction.
operation	varchar(255)	The operation performing the operation replay.

Column name	Column type	Description
isError	bit	Whether errors occurred during the synchronization or replay: 1 if errors were recorded, 0 if no errors were recorded.
isMonitored	bit	Whether the MBO is monitored. If 1, it is monitored. If 0, it is not.
isLogged	bit	Whether the domain is logging data. If 1, domain is logging data. If 0, it is not.

mms_mbs_message Table

Detailed history information for message-based synchronization.

Column name	Column type	Description
id	integer	The identifier of the data row.
deviceId	varchar(255)	The identifier of the physical device performing the operation replay.
userName	varchar(255)	The name of the user associated with the device ID.
packageName	varchar(255)	The package name of the mobile business object (MBO).
domain	varchar(255)	The domain to which the package involved in synchronization or operation replay belongs.
receiveTime	timestamp	The received time of inbound message. Not applicable to outbound messages.
pushTime	timestamp	The pushed time of outbound message. Not applicable to inbound messages.
startTime	timestamp	The date and time the message was initiated.
endTime	timestamp	The date and time the message was completed.

Column name	Column type	Description
processTime	integer	The total time of the message, in milliseconds.
mbo	varchar(255)	The MBO performing the message.
operation	varchar(255)	The operation performing the message.
messageType	varchar(50)	The type of message. One of: SUBSCRIBE, UNSUBSCRIBE, OPERATION_REPLAY, RECOVER, SUSPEND, RESUME, RESUMENOREPLAY, IMPORT_DATA, DATA_RESET, LOGIN, UNKNOWN_TYPE.
isError	bit	Value is 1 if errors were recorded during transaction. 0 if no errors recorded.
payloadSize	integer	The size of the message payload.
isPushMsg	bit	Value is 1 if the message is outbound, 1 if otherwise.
isRequestMsg	bit	Value is 1 if the message is inbound, 0 if otherwise.
isSubscription	bit	Value is 1 if the message is a subscription request, 0 if not.
isOperationReplay	bit	Value is 1 if the message is a message-based operation replay, 0 if not.
sentPayloadSize	integer	The payload size of the outbound message.
receivedPayloadSize	integer	The payload size of the inbound message.
isMonitored	bit	Value is 1 if MBO is monitored, 0 if not.

Column name	Column type	Description
isLogged	bit	Value is 1 if the domain is logging data, 0 if not.

mms_security_access Table

Information about security and user access.

Column	Column type	Description
id	integer	The identifier of the data row.
deviceId	varchar(255)	The identifier of the physical device used during the authentication request.
userName	varchar(255)	The name of the user requesting authentication.
packageName	varchar(255)	The package name of the authentication request.
domain	varchar(255)	The domain to which the package belongs.
securityConfiguration	varchar(255)	The name of the security configuration performing the authentication.
access_time	timestamp	The time the request for access was made.
outcome	bit	The outcome of the authentication request: 1 means authentication passed; 0 means authentication failed.
reason	longvarchar	Reason for authentication failure.

mms_rbs_outbound_notification Table

Outbound notification information for replication-based synchronization packages.

Column name	Column type	Description
id	integer	The identifier of the data row.

Column name	Column type	Description
deviceId	varchar(255)	The identifier of the physical device receiving the outbound notification.
userName	varchar(255)	The name of the user associated with the device ID.
packageName	varchar(255)	The package name of the MBO.
domain	varchar(255)	The domain to which the package belongs.
publicationName	varchar(255)	The synchronization group of the outbound notification.
notificationTime	timestamp	The time the outbound notification was sent.
subscriptionId	integer	The identifier for the subscription.
subcriptionEnabled	bit	Whether the subscription is enabled. If 1, it is. If 0, it is not.
isMonitored	bit	Whether the MBO is monitored. If 1, it is monitored. If 0, it is not.
isLogged	bit	Whether the domain is logging data. If 1, domain is logging data. If 0, it is not.

mms_data_change_notification Table

Information about data change notifications (DCNs) for messaging-based synchronization.

Column name	Column type	Description
packageName	varchar(255)	The package name of the mobile business object (MBO) affected by the DCN.
domain	varchar(255)	The domain to which the package involved in DCN belongs.
publicationName	varchar(255)	The synchronization group of the DCN.
notificationTime	timestamp	The time the DCN was sent.

Column name	Column type	Description
processTime	integer	The total time to process the DCN, in milliseconds.
affectedRows	integer	The rows affected by the DCN.
isMonitored	bit	Whether the MBO is monitored. If 1, it is monitored. If 0, it is not.
isLogged	bit	Whether the domain is logging data. If 1, domain is logging data. If 0, it is not.

mms_concurrent_user_info Table

Information about concurrent users.

Column name	Column type	Description
userName	varchar(255)	The name of the user associated with the device ID.
packageName	varchar(255)	The package name of the mobile business object (MBO) affected by the data change notification (DCN).
curTime	timestamp	The current time.
domain	varchar(255)	The domain to which the package involved in DCN belongs.
type	bit	The type of request. If 1, it is a messaging request. If 0, it is a replication request.

mms_queue_info Table

Information about the messaging queue.

Colum	Column type	Description
queueName	varchar(255)	The name of the queue.
pendingItem	integer	The number of pendings messages in the server queue.
curTime	timestamp	The current time.

mms_sampling_time Table

Information about the sampling time.

Column	Column type	Description
sampling_time	timestamp	The sampling time
id	integer	The unique key of the table

cache_history Table

Saves the history events on the Unwired Server cache by a deployed package.

Column	Column type	Description
package_name	varchar(128)	The package name of the mobile business object.
activity_type	integer	The event by activity type: • 1=ON DEMAND FULL REFRESH • 2=ON DEMAND PARTITIONED REFRESH • 3=CACHE QUERY
cache_name	varchar(128)	The Unwired Server cache name.
mbo_name	varchar(128)	The mobile business object that triggered the activity.
start_time	datetime	The recorded start date and time of the activity.
duration	bigint	The recorded duration of the activity.
partition_key	varchar(128)	The partition key value.
host_name	varchar(64)	The host name.
process_id	varchar(64)	The process id.
unique_row_id	numeric(10,0)	Internal identifier only.

cache_history Stored Procedures

The cache_history table uses several stored procedures.

Procedure	Parameters	Result
`get_lastfullrefresh`	• packagename varchar(128) • cachename varchar(128)	The last full refresh value.
`get_lastinvalidatetime`	• packagename varchar(128) • cachename varchar(128)	The last invalid date value.
`get_lastupdatetime`	• packagename varchar(128) • cachename varchar(128)	The last update value.

cache_statistic Table

Saves the Unwired Server cache status details.

Column	Column type	Description
package_name	varchar(128)	The package name.
cache_name	varchar(128)	The cache name.
last_full_refresh	datetime	The last date and time a full cache refresh occurred.
last_update	datetime	The last date and time a cache update occurred.
last_invalidate	datetime	The last date and time an invalidate cache occurred.

cache_statistics Stored Procedures

Provide aggregations of cache activities over a date range for a mobile business object.

Procedure	Parameters	Result
`get_package_mbo_maxfullrefreshtime` `get_package_mbo_minfullrefreshtime`	• @packagename varchar(128) • mboname varchar (128) • startdate datetime • endate datetime	The minimum or maximum value of the duration for the mobile business object over the start date and end date range for a refresh activity.

Procedure	Parameters	Result
`get_package_mbo_maxcachewaittime` `get_package_mbo_mincachewaittime`	• packagename varchar(128) • mboname varchar(128) • startdate datetime • endate datetime	The minimum or maximum value of the duration for the mobile business object over the start date and end date range for a cache activity.
`get_package_mbo_averageachewaittime` `get_package_mbo_averagefullrefereshtime`	• packagename varchar(128) • cachename varchar(128) • startdate datetime • endate datetime	The average value of the duration for the mobile business object over the start date and end date range for a cache or a refresh activity.
`get_package_mbo_access_count` `get_package_mbo_ondemandrefresh_count`	• packagename varchar(128) • cachename varchar(128) • startdate datetime • endate datetime	The count of access and on demand refresh activities for mobile businss object over the start date and end date range.

CHAPTER 15 Glossary: Sybase Unwired Platform

Defines terms for all Sybase Unwired Platform components.

administration perspective – Or administration console. The Unwired Platform administrative perspective is the Flash-based Web application for managing Unwired Server. *See* Sybase Control Center.

administrators – Unwired Platform users to which an administration role has been assigned. A user with the "SUP Administrator" role is called a "platform administrator" and a user with the "SUP Domain Administrator" role is called a "domain administrator". These administration roles must also be assigned SCC administration roles to avoid having to authenticate to Sybase Control Center in addition to Unwired Server:

- A domain administrator only requires the "sccUserRole" role.
- A platform administrator requires both the "sccAdminRole" and "sccUserRole" roles.

Adobe Flash Player – Adobe Flash Player is required to run Sybase Control Center. Because of this player, you are required to run Sybase Control Center in a 32-bit browser. Adobe does not support 64-bit browsers.

Advantage Database Server® – A relational database management system that provides the messaging database for Sybase Unwired Platform. *See* messaging database.

Afaria – An enterprise-grade, highly scalable device management solution with advanced capabilities to ensure that mobile data and devices are up-to-date, reliable, and secure. Afaria is a seperately licensed product that can extend the Unwired Platform in a mobile enterprise. Afaria includes a server (Afaria Server), a database (Afaria Database), an administration tool (Afaria Administrator), and other runtime components, depending on the license you purchase.

APNS – Apple Push Notification Service.

artifacts – Artifacts can be client-side or automatically generated files; for example: `.xml`, `.cs`, `.java`, `.cab` files.

BAPI – Business Application Programming Interface. A BAPI is a set of interfaces to object-oriented programming methods that enable a programmer to integrate third-party software into the proprietary R/3 product from SAP. For specific business tasks such as uploading transactional data, BAPIs are implemented and stored in the R/3 system as remote function call (RFC) modules.

BLOB – Binary Large Object. A BLOB is a collection of binary data stored as a single entity in a database management system. A BLOB may be text, images, audio, or video.

cache – The virtual tables in the consolidated database that store synchronization data. *See* CDB.

cache group – Defined in Unwired WorkSpace, MBOs are grouped and the same cache refresh policy is applied to their virtual tables (cache) in the CDB.

cache partitions – Partitioning the cache divides it into segments that can be refreshed individually, which gives better system performance than refreshing the entire cache. Define cache partitions in Unwired WorkSpace by defining a partition key, which is a load parameter used by the operation to load data into the cache from the enterprise information system (EIS).

CDB – Consolidated database. The CDB stores runtime metadata (for Unwired Platform components) and cache data (for MBOs). *See also* data tier.

CLI – Command line interface. CLI is the standard term for a command line tool or utility.

client application – *See* mobile application.

client object API – The client object API is described in the *Developer Guide for BlackBerry*, *Developer Guide for iOS*, and *Developer Guide for Windows Mobile*.

cluster – Also known as a server farm. Typically clusters are setup as either runtime server clusters or database clusters (also known as a data tier). Clustering is a method of setting up redundant Unwired Platform components on your network in order to design a highly scalable and available system architecture.

cluster database – A data tier component that holds information pertaining to all Unwired Platform server nodes. Other databases in the Unwired Platform data tier includes the consolidated, messaging, and monitoring databases.

connection – Includes the configuration details and credentials required to connect to a database, Web service, or other EIS.

connection pool – A connection pool is a cache of Enterprise Information System (EIS) connections maintained by Unwired Server, so that the connections can be reused when Unwired Server receives future requests for data.

connection profile – In Unwired WorkSpace, a connection profile includes the configuration details and credentials required to connect to an EIS.

context variable – In Unwired WorkSpace, these variables are automatically created when a developer adds reference(s) to an MBO in a mobile application. One table context variable is created for each MBO attribute. These variables allow mobile application developers to specify form fields or operation parameters to use the dynamic value of a selected record of an MBO during runtime.

data change notification (DCN) – Data change notification (DCN) allows an Enterprise Information System (EIS) to synchronize its data with the consolidated database through a push event.

data refresh – A data refresh synchronizes data between the consolidated database and a back-end EIS so that data in the cache is updated. *See also* scheduled data refresh.

data source – In Unwired WorkSpace, a data source is the persistent-storage location for the data that a mobile business object can access.

data tier – The data tier includes Unwired Server data such as cache, cluster information, and monitoring. The data tier includes the consolidated database (CDB), cluster, monitoring, and messaging databases.

deploy – (Unwired Server) Uploading a deployment archive or deployment unit to an Unwired Server instance. Unwired Server can then make these units accessible to users via a client application that is installed on a mobile device.

There is a one-to-one mapping between an Unwired WorkSpace project and a server package. Therefore, all MBOs that you deploy from one project to the same server are deployed to the same server package.

deployment archive – In Unwired WorkSpace, a deployment archive is created when a developer creates a package profile and executes the **build** operation. Building creates an archive that contains both a deployment unit and a corresponding descriptor file. A deployment archive can be delivered to an administrator for deployment to a production version of Unwired Server.

deployment descriptor – A deployment descriptor is an XML file that describes how a deployment unit should be deployed to Unwired Server. A deployment descriptor contains role-mapping and domain-connection information. You can deliver a deployment descriptor and a deployment unit—jointly called a deployment archive—to an administrator for deployment to a production version of Unwired Server.

deployment mode – You can set the mode in which a mobile application project or mobile deployment package is deployed to the target Unwired Server.

deployment profile – A deployment profile is a named instance of predefined server connections and role mappings that allows developers to automate deployment of multiple packages from Sybase Unwired WorkSpace to Unwired Server. Role mappings and connection mappings are transferred from the deployment profile to the deployment unit and the deployment descriptor.

deployment unit – The Unwired WorkSpace build process generates a deployment unit. It enables a mobile application to be effectively installed and used in either a preproduction or production environment. Once generated, a deployment unit allows anyone to deploy all required objects, logical roles, personalization keys, and server connection information together, without requiring access to the whole development project. You can deliver a deployment unit and a deployment descriptor—jointly called a deployment archive—to an administrator for deployment to a production version of Unwired Server.

development package – A collection of MBOs that you create in Unwired WorkSpace. You can deploy the contents of a development package on an instance of Unwired Server.

device application – *See also* mobile application. A device application is a software application that runs on a mobile device.

device notification – Replication-based synchronization (RBS) clients receive device notifications when a data change is detected for any of the MBOs in the synchronization group to which they are subscribed. Both the change detection interval of the synchronization group and the notification threshold of the subscription determine how often RBS clients receive device notifications. Administrators can use subscription templates to specify the notification threshold for a particular synchronization group.

device user – The user identity tied to a device.

DML – Data manipulation language. DML is a group of computer languages used to retrieve, insert, delete, and update data in a database.

DMZ – Demilitarized zone; also known as a perimeter network. The DMZ adds a layer of security to the local area network (LAN), where computers run behind a firewall. Hosts running in the DMZ cannot send requests directly to hosts running in the LAN.

domain administrator – A user to which the platform administrator assigns domain administration privileges for one or more domain partitions. The domain administrator has a restricted view in Sybase Control Center, and only features and domains they can manage are visible.

domains – Domains provide a logical partitioning of a hosting organization's environment, so that the organization achieves increased flexibility and granularity of control in multitenant environments. By default, the Unwired Platform installer creates a single domain named "default". However the platform administrator can also add more domains as required.

EIS – Enterprise Information System. EIS is a back-end system, such as a database.

Enterprise Explorer – In Unwired WorkSpace, Enterprise Explorer allows you to define data source and view their metadata (schema objects in case of database, BAPIs for SAP, and so on).

export – The Unwired Platform administrator can export the mobile objects, then import them to another server on the network. That server should meet the requriement needed by the exported MBO.

hostability – *See* multitenancy.

IDE – Integrated Development Environment.

JDE – BlackBerry Java Development Environment.

key performance indicator (KPI) – Used by Unwired Platform monitoring. KPIs are monitoring metrics that are made up for an object, using counters, activities, and time which jointly for the parameters that show the health of the system. KPIs can use current data or historical data.

keystore – The location in which encryption keys, digital certificates, and other credentials in either encrypted or unencrypted keystore file types are stored for Unwired Server runtime components. *See also* truststore.

LDAP – Lightweight Directory Access Protocol.

local business object – Defined in Unwired WorkSpace, local business objects are not bound to EIS data sources, so cannot be synchronized. Instead, they are objects that are used as local data store on device.

logical role – Logical roles are defined in mobile business objects, and mapped to physical roles when the deployment unit that contain the mobile business objects are deployed to Unwired Server.

matching rules – A rule that triggers a mobile workflow application. Matching rules are used by the mobile workflow email listener to identify e-mails that match the rules specified by the administrator. When emails match the rule, Unwired Server sends the e-mail as a mobile workflow to the device that matches the rule. A matching rule is configured by the administrator in Sybase Control Center.

MBO – Mobile business object. The fundamental unit of data exchange in Sybase Unwired Platform. An MBO roughly corresponds to a data set from a back-end data source. The data can come from a database query, a Web service operation, or SAP. An MBO contains both concrete implementation-level details and abstract interface-level details. At the implementation-level, an MBO contains read-only result fields that contain metadata about the data in the implementation, and parameters that are passed to the back-end data source. At the interface-level, an MBO contains attributes that map to result fields, which correspond to client properties. An MBO may have operations, which can also contain parameters that map to arguments, and which determines how the client passes information to the enterprise information system (EIS).

You can define relationships between MBOs, and link attributes and parameters in one MBO to attributes and parameters in another MBO.

MBO attribute – An MBO attribute is a field that can hold data. You can map an MBO attribute to a result field in a back-end data source; for example, a result field in a database table.

MBO binding – An MBO binding links MBO attributes and operations to a physical data source through a connection profile.

MBO operation – An MBO operation can be invoked from a client application to perform a task; for example, create, delete, or update data in the EIS.

MBO relationship – MBO relationships are analogous to links created by foreign keys in a relational database. For example, the account MBO has a field called *owner_ID* that maps to the *ID* field in the owner MBO.

Define MBO relationships to facilitate:

- Data synchronization
- EIS data-refresh policy

messaging based synchronization (MBS) – A synchronization method where data is delivered asynchronously using a secure, reliable messaging protocol. MBS provides fine-grained synchronization (synchronization is provided at the data level—each process communicates only with the process it depends on), and it is therefore assumed that the device is always connected and available. *See also* replication based synchronization.

messaging database – The messaging database allows in-flight messages to be stored until they can be delivered. This database is used in a messaging based synchronization environment. The messaging database is part of the Unwired Platform data tier, along with the consolidated, cluster, and monitoring databases.

mobile application – A Sybase Unwired Platform mobile application is an end-to-end application, which includes the MBO definition (back-end data connection, attributes, operations, and relationships), the generated server-side code, and the client-side application code.

Mobile Application Diagram – The Mobile Application Diagram is the graphical interface to create and edit MBOs. By dragging and dropping a data source onto the Mobile Application Diagram, you can create a mobile business object and generate its attribute mappings automatically.

Mobile Application Project – A collection of MBOs and client-side, design-time artifacts that make up a mobile application.

mobile workflow packages – Mobile workflow packages use the message-based synchronization model. The mobile workflow packages are deployed to Unwired Server, and can be deployed to mobile devices, via the Unwired Platform administrative perspective in Sybase Control Center.

monitoring – Monitoring is an Unwired Platform feature available in Sybase Control Center that allows administrators to identify key areas of weakness or periods of high activity in the particular area they are monitoring. It can be used for system diagnostic or for troubleshooting. Monitored operations include replication-based synchronization, messaging-based synchronization, messaging queue, data change notification, device notification, package, user, and cache activity.

monitoring database – A database that exclusively stores data related to replication and messaging synchronization, queues status, users, data change notifications, and device notifications activities. By default, the monitoring database runs in the same data tier as the consolidated database, messaging database and cluster database.

monitoring profiles – Monitoring profiles specify a monitoring schedule for a particular group of packages. These profiles let administrators collect granular data on which to base domain maintenance and configuration decisions.

multitenancy – The ability to host multiple tenants in one Unwired Cluster. Also known as hostability. *See also* domains.

node – A host or server computer upon which one or more runtime components have been installed.

object query – Defined in Unwired WorkSpace for an MBO and used to filter data that is downloaded to the device.

openDS – The default LDAP server that is installed in Developer Edition and is suitable for authentication and authorization in a development environment.

operation – *See* MBO operation.

package – A package is a named container for one or more MBOs. On Unwired Server a package contains MBOs that have been deployed to this instance of the server.

palette – In Unwired WorkSpace, the palette is the graphical interface view from which you can add MBOs, local business objects, structures, relationships, attributes, and operations to the Mobile Application Diagram.

parameter – A parameter is a value that is passed to an operation/method. The operation uses the value to determine the output. When you create an MBO, you can map MBO parameters to data-source arguments. For example, if a data source looks up population based on a state abbreviation, the MBO gets the state from the user, then passes it (as a parameter) to the data source to retrieve the information. Parameters can be:

- Synchronization parameters – synchronize a device application based on the value of the parameter.
- Load parameters – perform a data refresh based on the value of the parameter.
- Operation parameters – MBO operations contain parameters that map to data source arguments. Operation parameters determine how the client passes information to the enterprise information system (EIS).

personalization key – A personalization key allows a mobile device user to specify attribute values that are used as parameters for selecting data from a data source. Personalization keys are also used as operation parameters. Personalization keys are set at the package level. There are three type of personalization keys: Session, client, server.

They are most useful when they are used in multiple places within a mobile application, or in multiple mobile applications on the same server. Personalization keys may include attributes such as name, address, zip code, currency, location, customer list, and so forth.

physical role – A security provider group or role that is used to control access to Unwired Server resources.

Problems view – In Eclipse, the Problems view displays errors or warnings for the Mobile Application Project.

provisioning – The process of setting up a mobile device with required runtimes and device applications. Depending on the synchronization model used and depending on whether or not the device is also an Afaria client, the files and data required to provision the device varies.

pull synchronization – Pull synchronization is initiated by a remote client to synchronize the local database with the CDB. On Windows Mobile, pull synchronization is supported only in RBS applications.

push synchronization – Push is the server-initiated process of downloading data from Unwired Server to a remote client, at defined intervals, or based upon the occurrence of an event.

queue – In-flight messages for a messaging application are saved in a queue. A queue is a list of pending activities. The server then sends messages to specific destinations in the order that they appear in the queue. The depth of the queue indicates how many messages are waiting to be delivered.

relationship – *See* MBO relationship.

relay server – *See also* Sybase Hosted Relay Service.

replication based synchronization (RBS) – A synchronization method where data is delivered synchronously using an upload/download pattern. For push-enabled clients, RBS uses a "poke-pull" synchronization model, where a notification is pushed to the device (poke), and the device fetches the content (pull), and is assumed that the device is not always connected to the network and can operate in a disconnected mode and still be productive. For clients that are not push-enabled, the default synchronization model is pull.*See also* messaging based synchronization.

REST web services – Representational State Transfer (REST) is a style of software architecture for distributed hypermedia systems such as the World Wide Web.

RFC – Remote Function Call. You can use the RFC interface to write applications that communicate with SAP R/3 applications and databases. An RFC is a standalone function. Developers use SAP tools to write the Advanced Business Application Programming (ABAP) code that implements the logic of a function, and then mark it as "remotely callable," which turns an ABAP function into an RFC.

role – Roles control access to Sybase Unwired Platform resources. *See also* logical role and physical role.

role mapping – Maps a physical (server role) to a logical (Unwired Platform role). Role mappings can be defined by developers, when they deploy an MBO package to a development Unwired Server, or by platform or domain administrators when they assign a security configuration to a domain or deploy a package to a production Unwired Server (and thereby override the domain-wide settings in the security configuration).

RSOE – Relay Server Outbound Enabler. An RSOE is an application that manages communication between Unwired Serverand a relay server.

runtime server – An instance of Unwired Server that is running. Typically, a reference to the runtime server implies a connection to it.

SAP – SAP is one of the EIS types that Unwired Platform supports.

SCC – Sybase Control Center. A Web-based interface that allows you to administer your installed Sybase products.

scheduled data refresh – Data is updated in the consolidated database from a back-end EIS, based on a scheduled data refresh. Typically, data is retrieved from an EIS (for example, SAP) when a device user synchronizes. However, if an administrator wants the data to be preloaded for a mobile business object, a data refresh can be scheduled so that data is saved locally in a cache. By preloading data with a scheduled refresh, the data is available in the information server when a user synchronizes data from a device. Scheduled data refresh requires that an administrator define a cache group as "scheduled" (as opposed to "on-demand").

security configuration – Part of the application user and administration user security. A security configuration determines the scope of user identity, authentication and authorization checks, and can be assigned to one or more domains by the platform administrator in Sybase Control Center. A security configuration contains:

- A set of configured security providers (for example LDAP) to which authentication, authorization, attribution is delegated.
- Role mappings (which can be specified at the domain or package level)

security provider – A security provider and it's repository holds information about the users, security roles, security policies, and credentials used by some to provide security services to Unwired Platform. A security provider is part of a security configuration.

security profile – Part of the Unwired Server runtime component security. A security profile includes encryption metadata to capture certificate alias and the type of authentication used by server components. By using a security profile, the administrator creates a secured port over which components communicate.

server connection – The connection between Unwired WorkSpace and a back-end EIS is called a server connection.

server farm – *See also* cluster. Is the relay server designation for a cluster.

server-initiated synchronization – *See* push synchronization.

SOAP – Simple Object Access Protocol. SOAP is an XML-based protocol that enables applications to exchange information over HTTP. SOAP is used when Unwired Server communicates with a Web service.

solution – In Visual Studio, a solution is the high-level local workspace that contains the projects users create.

Solution Explorer – In Visual Studio, the Solution Explorer pane displays the active projects in a tree view.

SSO – Single sign-on. SSO is a credential-based authentication mechanism.

statistics – In Unwired Platform, the information collected by the monitoring database to determine if your system is running as efficiently as possible. Statistics can be current or historical. Current or historical data can be used to determine system availability or performance. Performance statistics are known as key performance indicators (KPI).

Start Page – In Visual Studio, the Start Page is the first page that displays when you launch the application.

structured data – Structured data can be displayed in a table with columns and labels.

structure object – Defined in Unwired WorkSpace, structures hold complex datatypes, for example, a table input to a SAP operation.

subscription – A subscription defines how data is transferred between a user's mobile device and Unwired Server. Subscriptions are used to notify a device user of data changes, then these updates are pushed to the user's mobile device.

Sybase Control Center – Sybase Control Center is the Flash-based Web application that includes a management framework for multiple Sybase server products, including Unwired Platform. Using the Unwired Platform administration perspective in Sybase Control Center, you can register clusters to manage Unwired Server, manage domains security configurations, users, devices, connections and monitor the environment. You can also deploy MBO packages and manage deployed MBO packages in order to design the synchronization behavior for those packages. Only use the features and documentation for Unwired Platform. Default features and documentation in Sybase Control Center do not always apply to the Unwired Platform use case.

Sybase Hosted Relay Service – The Sybase Hosted Relay Service is a Web-hosted relay server that enables you to test your Unwired Platform development system.

Sybase Messaging Service – The synchronization service that facilitates communication with device client applications.

Sybase Unified Agent – Provides runtime services to manage, monitor, and control distributed Sybase resources. The agent must be running for Sybase Control Center to run.

Sybase Unwired Platform – Sybase Unwired Platform is a development and administrative platform that enables you to mobilize your enterprise. With Unwired Platform, you can develop mobile business objects in the Unwired WorkSpace development environment, connect to structured and unstructured data sources, develop mobile applications, deploy mobile business objects and applications to Unwired Server, which manages messaging and data services between your data sources and your mobile devices.

Sybase Unwired WorkSpace – Sybase Unwired Platform includes Unwired WorkSpace, which is a development tool for creating mobile business objects and mobile applications.

synchronization group – Defined in Unwired WorkSpace, a synchronization group is a collection of MBOs that are synchronized at the same time.

synchronization parameter – A synchronization parameter is an MBO attribute used to filter and synchronize data between a mobile device and Unwired Server.

synchronization phase – For replication based synchronization packages, the phase can be an upload event (from device to the consolidated database) or download event (from the consolidated database to the device).

synchronize – *See also* data refresh. Synchronization is the process by which data consistency and population is achieved between remote disconnected clients and Unwired Server.

truststore – The location in which certificate authority (CA) signing certificates are stored. *See also* keystore.

undeploy – Running **undeploy** removes a domain package from an Unwired Server.

Unwired Server – The application server included with the Sybase Unwired Platform product that manages mobile applications, back-end EIS synchronization, communication, security, transactions, and scheduling.

user – Sybase Control Center displays the mobile-device users who are registered with the server.

Visual SQL – A graphical user interface tool that you can use to build SQL queries.

Visual Studio – Microsoft Visual Studio is an integrated development environment product that you can use to develop device applications from generated Unwired WorkSpace code.

Welcome page – In Eclipse, the first set of pages that display when you launch the application.

workspace – In Eclipse, a workspace is the directory on your local machine where Eclipse stores the projects that you create.

WorkSpace Navigator – In Eclipse, the tree view that displays your mobile application projects.

WSDL file – Web Service Definition Language file. The file that describes the Web service interface that allows clients to communicate with the Web service. When you create a Web service connection for a mobile business object, you enter the location of a WSDL file in the URL.

Index

A

B

C

D

E

F

G

H

I

J

K

L

M

N

O

P

Q

R

S

T

U

V

W

X